THE UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR Daniel T. Johnson

TITLE OF THESIS Decision Theory and Automatic Planning

DEGREE FOR WHICH THESIS WAS PRESENTED Master of Science

YEAR THIS DEGREE GRANTED Fall, 1977

THE UNIVERSITY OF ALBERTA

DECISION THEORY AND AUTOMATIC PLANNING

by

(C) DANIEL T. JOHNSON

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE

OF MASTER OF SCIENCE

DEPARTMENT OF COMPUTING SCIENCE

EDMONTON, ALBERTA

FALL, 1977

THE UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research, for acceptance, a thesis entitled ....

..Decision Theory and Automatic Planning..........................

....

submitted by ..Daniel T. Johnson..................................

in partial fulfilment of the requirements for the degree of Master of Science.

ABSTRACT

A traditional aim of planning systems is the optimal trade-off of cost of planning and cost of execution. Such optimization of cost is the subject of classical statistical decision theory; a synthesis seems called for.

The method of classical decision theory is to model a problem with a decision tree, then to compute expected costs by backward induction. This cannot be used here; building the tree is *doing* the planning. A compromise is proposed, of using a decision tree only one level deep, and then using cost estimates for the tip nodes. This is based on traditional Artificial Intelligence work on mini-max game playing, A* search, and MULTIPLE; however, this proposal uses changes in both probability and expected cost. A control structure in the style of MULTIPLE is developed, unfortunately requiring a number of ad hoc steps.

A hand-simulated STRIPS-like robot is described, using a simplified form of the proposed decision method. The workability and advantage of the method are demonstrated. Hope is offered for the ultimate development of a theory of optimal planning free of ad hoc reasoning.

ACKNOWLEDGEMENTS

I wish to thank my supervisor, Professor L. K. Schubert, for his many hours spent discussing this work and advising me on it.

I thank my wife, Diane Persson, without whose kind cooperation and firm encouragement this thesis might never have been completed.

The financial assistance received from the National Research Council of Canada is gratefully acknowledged.

TABLE OF CONTENTS

CHAPTER PAGE

CHAPTER PAGE

LIST OF FIGURES

LIST OF FIGURES (Cont.)

FIGURE		PAGE

LIST OF TABLES

TABLE | PAGE

CHAPTER 1

INTRODUCTION

A frequently cited desirable feature of a planning system is that it should optimally trade off cost of planning and cost of execution. However, how this optimization is to be achieved is seldom specified; usually merely the admonition to "keep this in mind when writing programs" is given. An explicit framework for achieving such optimality is provided by statistical decision theory. The possibility of applying decision theoretic methods to traditional Artificial Intelligence planning systems inspired the present work.

The implication is the use of a numerical measure of worth (utility) to make the decisions involved in producing a plan. This idea is not new nor peculiar to a decision theoretic approach. The best known search algorithm of Artificial Intelligence, A*, has this form, as does mini-max game playing and the theorem prover MULTIPLE.

Work of a specifically decision theoretic motivation is much less common. Significant examples include the PERCY simulation of Jacobs and Kiefer [6], the JASON robot of Coles et al. [1], and particularly the work of Feldman [4,3]. Feldman's work especially was an inspiration to

this thesis, particularly in the application of decision theory to a STRIPS-like planning system. This related work is reviewed in Chapter 2.

A very desirable feature of a planning system is the ability to back up, that is, to reconsider an alternative previously bypassed, when the initial choice fails (or merely becomes less attractive). This is dealt with only in cursory fashion by the other workers. Feldman deals only with outright failure, and this only in passing. Including reconsideration explicitly in a planning tree (as Coles et al. [1] do with JASON) quickly complicates the tree. For example, consider a simple planning tree with two alternative plans:

```
                ____
 __________| OR |____________
|          |____|            |
|                            |
PLANA                      PLANB
```

If PLANA has a cost of 4 and PLANB of 6, PLANA is the obvious choice. Suppose, however, PLANB really consists of two steps, the initial step costing 1 (after which we may reconsider) and specifying the cost of the second step as 10 (with probability .5) or 0 (with probability .5). Then while the expected cost of PLANB is still 6, it is not the best choice. A complete decision tree (Figure 1) shows why; if it results that step two of PLANB will cost 10, we reconsider PLANA and choose it, at a cost of only 4. Thus allowing for reconsideration, PLANB has an expected cost

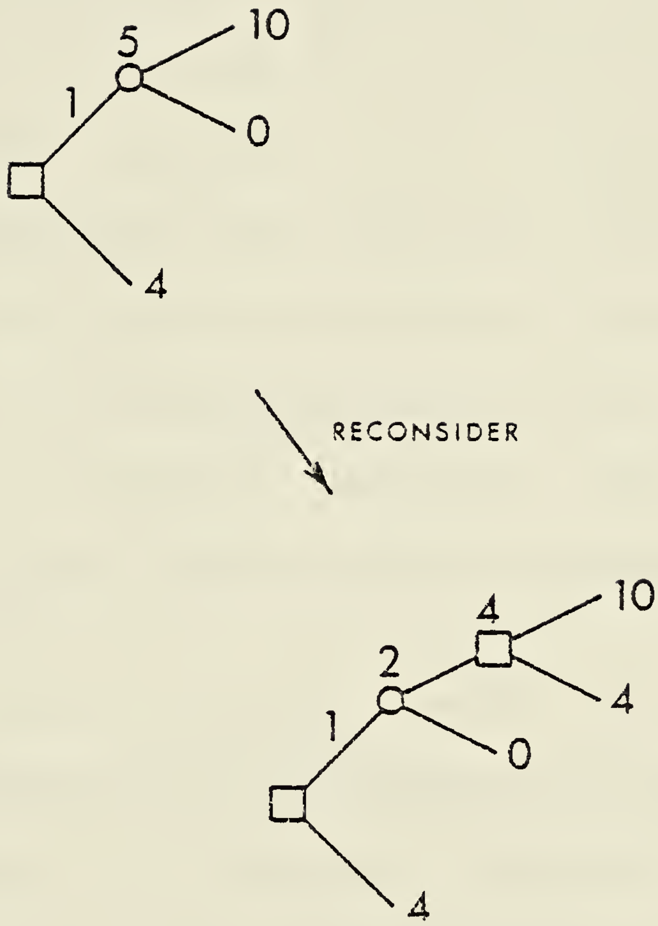


FIG. 1 RECONSIDERATION

of only 3. This is treated more fully in Section 3.2.

If an alternative may have more than two steps, that is, more than one point where more accurate knowledge of remaining cost is acquired, things grow exponentially more complex. Compounding this with already complex planning trees is clearly infeasible in general. An approximation is needed, to allow us to include the significant effects of reconsideration without such complexity. The approximation investigated here limits an alternative to two steps, for the purpose of making a choice. Thus the first order effects of reconsideration will be handled properly, hopefully yielding adequate decision making behavior when the higher order effects are small.

Such drastic forward pruning would be expected to give severe degradation of performance in a game-playing environment. However, a planning structure should be <u>designed</u> to make the most significant information the first available; this should limit the degradation in the planning environment. In fact, for some problems, one level of expansion is totally adequate; such an example is discussed in Section 3.1.

This results in a planning system where alternatives are characterized as two step processes. The first step has an expected cost termed s, after which more precise information about expected remaining cost (second step cost, termed t' and probability of success (termed p')

become available. These refined estimates obey a bivariate distribution function $f(t',p')$. The numerical characterization of an alternative consists then of the number s and the distribution f. This is treated fully in Section 3.3. In a practical implementation, some numerical approximation of the distribution function $f(t',p')$ must be used. Possibilities are discussed in Section 3.4.

The control structure then would be like that of MULTIPLE. As expansion of a top node produces a new characterization for the tip process, this characterization is "backed up" through the ancestral AND and OR nodes to the root; choices along the way are re-evaluated. How this backing up is accomplished is the subject of Section 3.5.

The proposed characterization approximates all processes by two step processes. Multi-step processes may be partitioned into two steps in many ways. Criteria for choosing a partitioning are discussed in Section 3.6.

It is possible for processes invoked in planning to run on forever in an attempt to achieve a goal, which may result in the expected cost of the process being infinite. This may be so even of a desirable process; for example, one which succeeds with zero cost 99 per cent of the time, and goes on forever the remaining 1 per cent. These cases are covered in Section 3.7.

The proposed one step look-ahead decision method is worked out in a detailed example in Chapter 4. A STRIPS-like robot is used. In making the descent from the lofty theoretical heights of Chapter 3 to the real world, considerable simplification is necessary, the principal one being to use an estimate for probability of success which is uniformly 1. Thus the distribution of revised cost f(t',p') becomes a simple one dimensional function f(t'). Execution of the proposed planning system is elaborated fully by hand for several example goals. These demonstrate the advantage of the proposed look-ahead method over simple expected cost methods.

While even with these simplified examples the worth of the proposed planning method remains, a number of details of implementing the full proposal are unspecified, and remain open questions. These difficulties and possibilites are discussed in the concluding chapter.

CHAPTER 2

RELATED WORK

2.1 MULTIPLE

Previous systems have endeavored to use quantitative measures of worth to guide planning. Of these, the one most directly related to the current work is Slagle's MULTIPLE [12]. Here, probability of success was the sole measure of worth, and remaining cost was approached only indirectly, as rate of change of probability with effort. The big advantage is that probability can be "backed up" in a natural way over the logical connectives AND and OR in a problem reduction tree. Logical extension allows the same for the derivatives

$$P[X \text{ AND } Y] = P[X]P[Y]$$

$$P[X \text{ OR } Y] = 1 - (1-P[X])(1-P[Y])$$

$$\frac{d}{dt}P[X \text{ AND } Y] = P[X]\frac{d}{dt}P[Y] + P[Y]\frac{d}{dt}P[X]$$

$$\frac{d}{dt}P[X \text{ OR } Y] = (1-P[X])\frac{d}{dt}P[Y] + (1-P[Y])\frac{d}{dt}P[X]$$

where X and Y are subgoals. This is also done for change of goal probability for effort T_X invested in subgoal X

$$\left|\frac{d}{dT_X} P[X \text{ AND } Y]\right| = \left|\frac{d}{dP[X]} P[X \text{ AND } Y]\right| \frac{d}{dT_X} P[X]$$

$$\left|\frac{d}{dT_X} P[X \text{ OR } Y]\right| = (1-P[Y]) \left|\frac{d}{dT_X} P[X]\right|$$

The merit of working on any subgoal was defined as the rate of change of top-goal probability for effort invested in that subgoal. The (self) merits and probabilities of tip subgoals were heuristically evaluated (actually estimated by a Learning Program, but this is not relevant here). The merit of higher level nodes was simply the maximum merit among successors, since the "choice" subgoal to attack next would be that of greatest merit. At each elaboration, these parameters of probability, merit, and choice could be backed up to the top node by considering only those nodes on the path to the node elaborated, and their sibling nodes. Thus, the cost of evaluating a node grows only linearly with the depth of the node; this is a big advantage of MULTIPLE, and a feature desirable to retain in any extension of the system.

However, this is intimately associated with a most frequent criticism of MULTIPLE: the assumption of independence among subgoals. The naivete of such an assumption about conjoined subgoals in a robot problem solver has been more than adequately demonstrated by Sacerdoti [8] in ABSTRIPS and Siklossy [10] in LAWALY. Here, the idea of "remaining freedom" (Siklossy)

determines the ordering of conjuncts. This points up a significant difference between the dynamic world of a robot problem solver, and the static theorem proving model which is the basis of MULTIPLE. Contrary to the notion of PLANNER, there _is_ a major difference between proving a proposition and achieving a goal. The difference arises because achieving a subgoal may alter the state of the world relevant to achieving (or proving) the next subgoal. Because working on "earlier" subgoals increases the specificity of later subgoals, problem reduction leads to highly interdependent subgoals. Thus the derivative of probability (and cost) of a subgoal with respect to work performed on other subgoals is (generally) nonzero. A good example is the goal ((ROBOT INROOM D) AND (BOX1 NEXTO BOX2)) discussed in relation to STRIPS in Section 2.6.1.

There are ways of limiting the action of MULTIPLE or similar strategies to avoid some of these interdependence problems; these are discussed elsewhere in relation to the application of my proposal. However, there is another problem with MULTIPLE in a dynamic environment. Consider a world in which any legal goal is achievable. It is still possible in such a world for some solutions to be shorter than others, or easier to find: we can do better than random choice. How can MULTIPLE help us solve problems _efficiently_? The merit of every subgoal will be zero, that is, MULTIPLE provides no guidance at all. Clearly, investment of effort can have salutary effects other than

altering probability of success. Ignoring this fact is perhaps the greatest deficiency of MULTIPLE.

2.2 PERCY

The evolution of a planning process toward a particular goal can be modelled as a game against the "environment". The planner makes decisions among a set of possibilities. At alternate moves, the "environment" specifies outcomes, much like an opponent in a chess game except that this opponent is random rather than fighting back. This model has been explored by Jacobs and Kiefer [6] with their PERCY simulation. PERCY is a simple-minded insect with a single task: building a nest. In PERCY's world, there are a total of only 12 states (termed outcomes), and for each outcome, there is specified a set of legal decisions (cardinality ≤ 4) which PERCY can make. There is a utility measure on the outcomes, and a probability distribution over the set of outcomes for each legal outcome-decision pair. These utility distributions are learned by a Bayesian update of estimates.

A complete optimal decision method would be to generate a complete tree of all possibilities to the tip nodes of task completion, where the utilities of outcomes are trivially known. Then, the utilities of higher outcomes and decisions can be calculated by backward induction: the utility of an outcome is the maximum of the

utilities of successor decisions, and the utility of a decision is the expected value of the successor outcomes. This is in contrast to the mini-max backup of usual game playing; the authors point out that mini-max backup is a special case where the probability distribution of outcomes is such that the minimum utility has probability 1. In PERCY's world, the tree is infinite, so this cannot be done; but like other situations, the planner in fact expands the tree to a limited depth, and uses utility estimates at tip nodes. In PERCY's case, the depth of expansion is one -- only a single step of look-ahead. That is, it evaluates

$$u(d) = \sum_{\text{outcomes}} P[\text{outcome} \mid d]\, u(\text{outcome})$$

the expected utility of decision d, for all decisions, and makes the decision of maximum utility.

The tremendously simplified type of problem solver generated to illustrate this approach shows its deficiencies. It is a state space formulation (as opposed to problem reduction), and only a simple world can be modelled by a workable number of states. Like MULTIPLE or A*, PERCY relies on complete elaboration of a goal to make its utility estimates. Despite these simplifications, it seems that one level of look-ahead is all that is practical. However, the adequacy of such a myopic strategy is demonstrated on PERCY's tasks which usually require

many steps -- another example of locally determined behavior satisfying global goals.

2.3 JASON

Coles et al. [1] have incorporated decision analysis methods in their JASON robot. They note the connection with Feldman's work:

> In a more recent paper by Feldman and Sproull this same problem (the monkey and bananas problem) has served as the basis for a decision-theoretic approach. Although they are much more concerned with the use of a numerical utility function during planning to develop a more efficient strategy, we have independently reached the same conclusion regarding the positive value of joining decision analysis with a symbolic robot problem solver to facilitate intelligent decision making under conditions of uncertainty. . . . In particular, using this approach, we can create a decision tree that allows one to "roll back" the consequences of further information gathering operations in comparison with direct action.

In JASON's case, "information gathering" is limited to exploration of the external environment, and does not allow for elaboration of new information through planning. An even more serious deficiency is revealed by the assignment of costs:

> Now in this formulation, although "thinking" is _assumed to be free of charge,_ every operation JASON can carry out in the real world is energy (cost) consuming.

Of course, when thinking is free, one can use classical decision theory directly: consider all consequences of all decisions, and compute the utilities by backward induction. This is what JASON does. Fortunately, for the

world and goal discussed, the resulting tree has only 46 nodes, a direct consequence of the simple formulation. In a more realistic world, of course, the complexity of this method explodes hopelessly; thinking, no matter how cheap, can never compensate for exponentially proliferating alternatives in an exhaustive search. So while these authors have explored some of the advantages of decision theory, they have avoided a central question: how to plan optimally, taking into account the cost of planning. Precisely this question prompted the present work.

2.4 A*

The idea of using a measure of remaining cost to guide problem solving search is certainly not new; the A* algorithm of Hart, Nilsson, and Raphael [7] is a prime example. As first introduced, it uses a state-space representation of the problem solving environment. However, extension to a problem reduction framework is not difficult, and has also been pursued by Nilsson [7]. Major differences are the inclusion of AND nodes, and a conceptual reversal to proceed backwards, producing reduced goals from previous ones until they are trivially satisfied.

Further, in a STRIPS-like world, it is easy to identify a set of predicate calculus assertions with the idea of a "state". However, a difficulty results because a

goal is not a simple state, but a set of states satisfying the goal description. While there is no theoretical problem in introducing suitable equivalence classes, a practical problem arises in devising suitable heuristic cost estimates. This is precisely the problem which surfaces in the snowshovel search (Section 3.1): we need cost estimates for parameterized plans.

There is another problem with direct application of A*. It proceeds by expanding a node fully, then choosing a minimum cost successor. The inadequacy of such an approach is obvious when the number of successors is large, or infinite. Full expansion is unfortunately a necessary feature of A*, since it is essential to A*'s "admissibility". Rather, this full expansion is a necessary result of the admissibility requirement that A*'s heuristic underestimate true cost. In the case of a parameterized goal where cost may have a continuous distribution, the probability of any instance achieving the minimum is zero, so we expect full expansion to take place. Clearly, we must abandon admissibility in favor of optimality. My proposal can be viewed as an extension of A* in the sense of using a heuristic to evaluate remaining cost. However, a simple remaining-cost measure is inadequate, as I argue in Section 3.2.

2.5 Feldman's work

2.5.1 Image processing

Advocacy of applying decision theory to problem solving in Artificial Intelligence is rare: the work of Feldman is an exception and must take much credit for inspiration of the current work. Arguments in favor of applying decision theory to Artificial Intelligence are cited by Feldman and Yakimovsky [4].

> . . . choice between alternative courses of action is often inherently numerical. One chooses the cheapest, fastest, strongest, etc. alternative.

> A related point is the prevalence of probablistic judgements in the world.

> . . . 'comparing the incomparable'. If flying is faster and safer than driving, but more expensive and subject to delay, how can we choose which to do?

> In the absence of a UF [utilities function], a heuristic program apparently must have rules covering all possible combinations of goals and circumstances. The addition of new entities will require significant reprogramming. . . . Decision theory provides a uniform way of treating information related to choosing a course of action, given the relevant utility and probability values.

I find this a most convincing argument, an argument against procedural embedding of decision theory. It is, unfortunately, not applied in their picture partitioning

system.

> The other basic method, [other than hierarchical planning] for planning is to attempt to measure progress toward a goal and to mix planning with action. Obviously enough, a uniform measure of progress toward a goal involves a UF.

This view of a utility function is in the spirit of A*, but I shall argue that it must be combined with hierarchical planning to be practical.

Feldman and Yakimovsky [4] describe application of decision theory to the problem of segmenting an image into meaningful regions. The goal was a front-end region analyzer (therefore fast and unsophisticated) which was still independent of absolute criteria (example: boundary strength). The chosen utility function was the probability of a correct interpretation; (an interpretation is a partition and a labelling of each region). Regions are "grown" by merging 2 adjacent regions at each iteration; basic decisions are which regions merge, and when to stop. At each iteration, upper and lower bounds on the probability (utility) are computed. When these begin to decrease, the system stops growing and assigns an interpretation to the partition of highest utility.

Despite all the talk of decision analysis, this system does not seem to follow the usual form of elaborating successions of alternating decisions and probablistic outcomes. No backing up of utilities is used. Rather, the utility function idea is used as an

inspiration for an A*-like heuristic of remaining cost (actually the inverse of remaining cost; the goal is achieved when probability is maximized rather than remaining cost minimized). The problem representation is certainly of the state-space variety: we are concerned with transitions involving merging two regions. So, while the arguments the authors cite in favor of using a decision theory approach are convincing, their example does not demonstrate an explicit application.

2.5.2 Monkey and bananas problem

Feldman and Sproull [3] have investigated many of the aspects of applying decision theory to problem solving through consideration of a robot symbolic problem solver. They discuss a STRIPS type robot solving the monkey and bananas problem, giving a solution like WALKTO (BOX), PUSHTO (BOX,BANANAS), CLIMB(BOX), CONSUME(BANANAS). They point out that

> . . . the 'plan' itself is trivial. . . . It is the application of this plan to the world situation which is difficult. We believe that much intelligent activity is characterized by complex applications of simple plans and this belief has led us to concentrate on the closely related questions of plan elaboration and execution.

In this context, plan elaboration is the choice among several alternative instantiations of BOX. The approach is to evaluate the utilities of the alternatives, and select the one of highest utility:

> Because the utility of a plan can be used to compare the merits of competing plans, it can be used to guide a search for good plans. The basic idea is to search by expanding paths of greatest expected utility.

I criticize this approach elsewhere in this thesis.

The authors are not unaware of the inadequacy of using expected cost only; they talk of using bounds on cost to prune planning trees (an extension of A*'s lower bounding). This approach also has obvious shortcomings.

They consider the possibility of planning to recover from various "failures", and point out that

> This process can be carried on indefinitely, but . . . the cost of additional planning may exceed the slight improvement in expected utility.

A similar issue is the insertion of information gathering steps, and in general any kind of "procedural" planning, where some decisions must be made at execution. Instead of elaborating a plan for failure recovery, they propose

> . . . an estimate of the utility of fixing up the failure is the current utility measurement of the top-level alternative.

I suggest use of this substitution of alternatives in situations other than outright failure. While the authors note this interdependence of utilities (" . . . the vision and action planning activities each make use of the other's estimates.") they do not comment on resolving the implicit deadlock. Rather, they merely note the more general problem:

> Unfortunately, specifying a utility function that reflects the benefits of future planning [the aim of this thesis] is quite difficult. In fact this is the classic 'cost of analysis' problem in decision theory: if a decision analysis must include the cost of the analysis, we are led to a recursive, unterminating computation: what is the cost of analyzing the cost of analysis? Thus it appears 'impossible' to compute the cost of decision analysis.

Fortunately, this does not make the implementation of decision analysis "impossible".

2.6 The STRIPS-like problem solvers

An environment for problem solving has been developed which illustrates many of the general difficulties, yet is sufficiently simple to be tractable. This is the blocks world of STRIPS, ABSTRIPS, and LAWALY. The world is described by assertions in a predicate calculus format (or equivalently a semantic network); this paradigm holds greatest promise of a uniform representation of knowledge. The basic world consists of a robot in a building with a number of rooms and doors; this basic shell is usually populated with other objects: boxes, lightswitches, keys, etc. Any operator available to the robot is described as an operator schema: generally, a statement of *preconditions* describing worlds to which the operator can be applied, and specifications of *transformations* describing the corresesonding resulting worlds. The implementations specify transformations by delete and add lists, of assertions to be deleted from and added to the

world model.

Many classical Artificial Intelligence problems can be cast in this model. For example, Feldman gives a representation of the monkey and bananas problem. This is the framework in which the current proposal was conceived to operate.

2.6.1 STRIPS

The earliest of the above systems was STRIPS [5]. Ignoring its use of resolution principle for inference, STRIPS regards an operator relevant to a goal if its add list includes one of the desired goal literals. That literal is then replaced by the precondition(s) of the operator, expanded until the subgoals thus generated are true in the initial world. When this happens, the operator is applied in the sense that the world description is updated; this new world is the "initial" world for other planning branches. This depth first search behavior is not explicitly required of STRIPS; it is merely the usual result of the method used to select the next goal literal to be expanded. However, if search is not depth-first, considerable wasted effort may result. For example, consider the world of Figure 2, and the goal ((ROBOT INROOM D) AND (BOX1 NEXTO BOX 2)). Clearly, if the planning to achieve the two goals proceeds in parallel, the plan to go from room C to room D (at least as far as

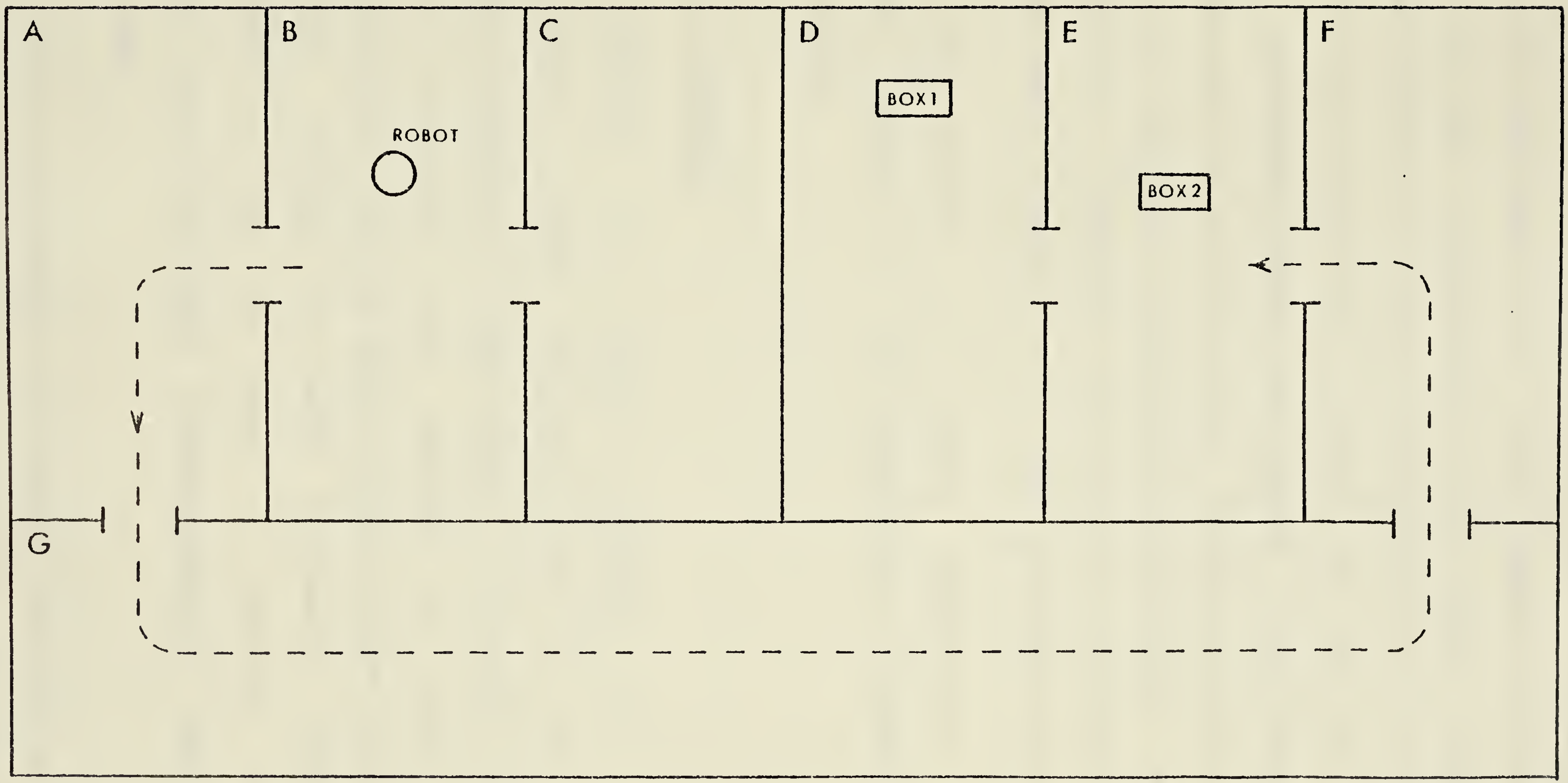


FIG. 2 DUPLICATION POTENTIAL OF STRIPS

room E) will be generated twice: both conjuncts seek a path from the same initial world. However, transformations must be made sequentially; the initial world for the second conjunct is always the resultant of the achievement of the first. Even if it is known that the first conjunct must be achieved before the second, we may have a problem in planning the second conjunct before planning the first: the world resulting from achieving the first conjunct is incompletely specified. And there is a further constraint on the second conjunct -- it must not undo the first.

These problems are handled differently (but with similar results) by two refinements of STRIPS: ABSTRIPS and LAWALY.

2.6.2 ABSTRIPS

ABSTRIPS [8] orders conjuncts by assigning "criticality levels" to each literal. It then solves the problem completely at each level, by ignoring literals of lower criticality. This ignoring of details is termed "abstraction", and the whole process termed "hierarchical" planning. Because it avoids obviously false steps, ABSTRIPS is about an order of magnitude faster than STRIPS, and can solve correspondingly more complex problems.

These are somewhat misleading terms for so simple an operation as throwing out details. Indeed, abstraction and

hierarchy do not in general reduce to ignoring detail. Rather, they seem to correspond more closely to transitions between whole systems of operators and concepts. For example, STRIPS and other robot problem solvers work with higher level operators; given a plan as a sequence of such operators, for example GOTO (BOX1), PUSHTO (BOX2), the real robot must translate this into a lower order plan to operate stepping motors on its wheels or whatever. Sacerdoti's later work on NOAH [9] gives a better general framework for hierarchy and abstraction.

2.6.3 LAWALY

LAWALY [10] demonstrates that the same kind of ordering of operators can be obtained in a conceptually more straight-forward manner. For example, given the world of Figure 3, and the goal ((BOX1 INROOM A) AND (ROBOT INROOM C)) it is clear that (BOX1 INROOM A) should be achieved first. Siklossy characterizes this as remaining freedom: protecting the assertion of some literals prevents the achievement of others. In this example, once the robot's position is fixed, it is no longer free to move boxes.

Conjoined literals can be ranked by achieving last that conjunct most easily achieved, assuming the others are already true. By thus ordering conjuncts (done for preconditions only once, in advance), avoiding inference

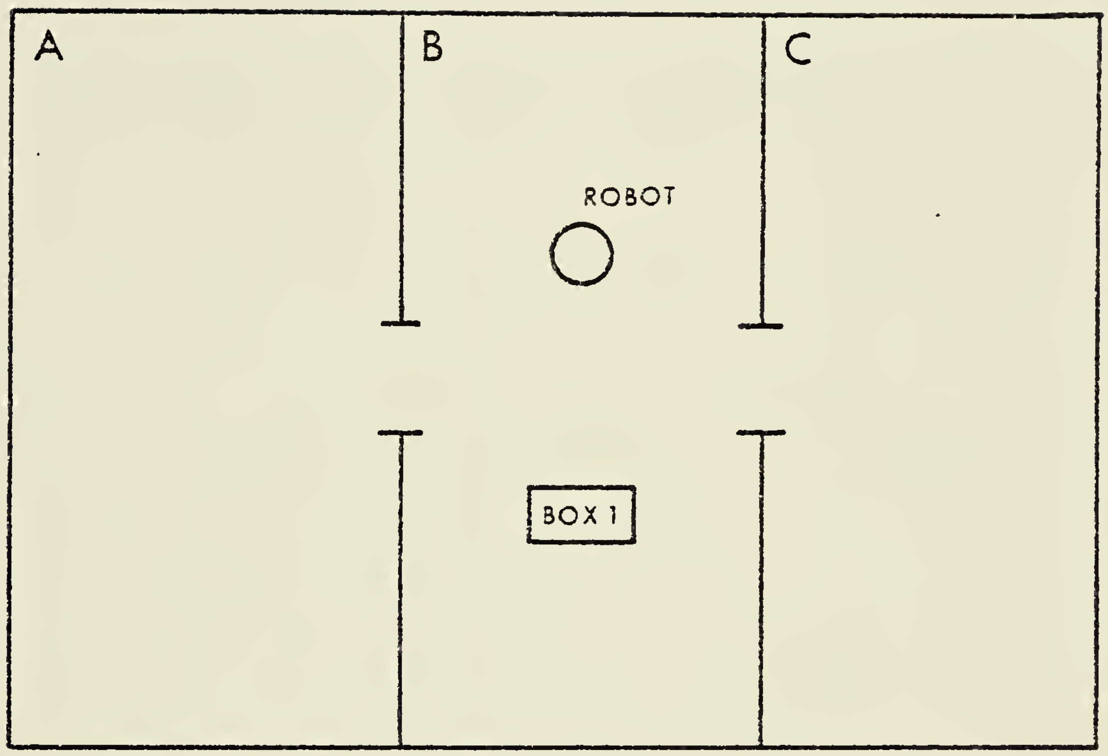


FIG. 3 LINEARIZING GOALS

(which I haven't observed to aid STRIPS in any meaningful way), and using a domain specific algorithm to find routes between rooms, instead of blind search, LAWALY achieves speeds about two orders of magnitude faster than STRIPS, and can solve problems requiring hundreds of steps.

CHAPTER 3

LOOK-AHEAD DECISIONS

3.1 A Sequential Search Example:

The Monkey and Bananas Problem, a la Feldman

OR

How I Found my Snowshovel in the Want Ads

To illustrate the probelms of including planning cost in the cost of a plan, we shall examine a particular example in detail. The example is drawn from the formulation of the monkey and bananas problem considered by Feldman and Sproull [3], and has the advantage that theory is adequate to analyze it fully (it is an instance of the entrance fee problem discussed by DeGroot [2]), to allow comparison with my proposal of a one level look-ahead approximation.

Suppose, as in Feldman, that a monkey has obtained a parameterized solution to the classic culinary problem, say (following Feldman):

$$\left.\begin{array}{l} \text{WALKTO(BOX)} \\ \text{PUSHTO(BOX,BANANAS)} \\ \text{CLIMB(BOX)} \\ \text{CONSUME(BANANAS)} \end{array}\right\} \quad \text{PLANA(BOX)}$$

where we consider BOX to be a parameter to be

instantiated. Let us further assume that the cost of executing the plan is proportional to the distances between the monkey and the box, and between the box and the bananas; if we have initially the monkey at the same location as the bananas, these are the same. Now, let the monkey and bananas be in the center of a circular room of radius H, and containing an arbitrarily large number of boxes, uniformly distributed. This could be represented as an OR node:

```
                       ____
     __________________| OR |____________________   . . .
    |                  |____|                    |
    |                     |                      |
PLANA(BOX1)         PLANA(BOX2)
```

However, all successors cannot be enumerated. This is the fatal flaw of evaluation functions which work only on fully instantiated plans. What we really need is something that can make the choices one at a time:

```
                    ____
        ____________| OR |________________
       |            |____|                |
       |                                  |
PLANA(BOX1)                      PLANA(BOX | not BOX1)
```

Now, if we are to follow Feldman's suggestion, we need to evaluate the costs (utilities) of the alternatives. This is easy for the fully instantiated PLANA(BOX1); it is simply

$$C_1 = az_1$$

where z_1 is the distance of BOX1 from the center of the room and a is the constant of proportionality. But what about the cost of the alternative? Clearly, this depends on the decision strategy used. For example, if we adopt the strategy of evaluating all the alternatives, and choosing the least expensive, we have run-away cost if we assign any non-zero cost to enumerating each alternative. Let us assume unit cost for each such elaboration. That is, we have incurred unit cost in going from:

PLANA(BOX)

to:

```
                 ____
        ________| OR |______________
       |        |____|              |
       |                            |
  PLANA(BOX1)                  PLANA(BOX | not BOX1)
```

Now, to choose between these alternatives, our strategy is to choose the one of least cost. But the cost of PLANA(BOX | not BOX1) (like the cost of PLANA(BOX)) is dependent on our strategy. Our reasoning is locked in an infernal loop.

Let us consider a more realistic problem whose structure is identical to that of the many-box monkey and bananas problem. Its advantage is that the cost of enumerating alternatives is plausibly and simply defined. Suppose we wish to purchase a used snowshovel through the want ads. Living in the center of a large city, with an

arbitrarily thick newspaper (like the Edmonton Journal), we assume a uniform spatial distribution of an arbitrarily large number of snowshovels-for-sale. We have set aside sufficient cash, and valuing our time, wish to obtain a shovel as quickly as possible. Having located a shovel, the time it takes to get it will be proportional to the distance to the vendor. We assume the ads include no addresses; locating a shovel will consist of finding the next shovel-ad, telephoning, and determining the location. Assume that locating the next shovel takes unit time (cost), and that the city has unit radius. Now, the distribution of shovels by distance will be the marginal $f(x)$ of a uniform density function $g(x,\theta)$ on the unit circle (see Figure 4). Let $F(x)$ be the cumulative distribution function corresponding to $f(x)$. At any time, we may stop locating shovels and go buy one; obviously, the minimum cost will correspond to the closest one. So, we can look at our search as a process defined by one parameter, z, the distance to the closest shovel found so far. At the next step, the process may evolve to a lower z, distributed like $f(x)$, or stay the same if the next shovel is no closer than the nearest (see Figure 5).

Now, our decision strategy will consist of a stopping rule: determining those values of z for which we should stop the process and obtain the shovel at distance z. Clearly, these values of z will be a set of the form $\{z: z \leq T\}$ (since we can always transform a lower cost plan into

$$g(x,\theta) = \frac{1}{\pi} \qquad 0 \leq x \leq 1\ ,\ 0 \leq \theta < 2\pi$$

$$F(x) = \int_0^x \int_0^{2\pi} g(t,\theta)\, t\, d\theta\, dt$$

$$= \int_0^x 2t\, dt = x^2$$

$$f(x) = 2x$$

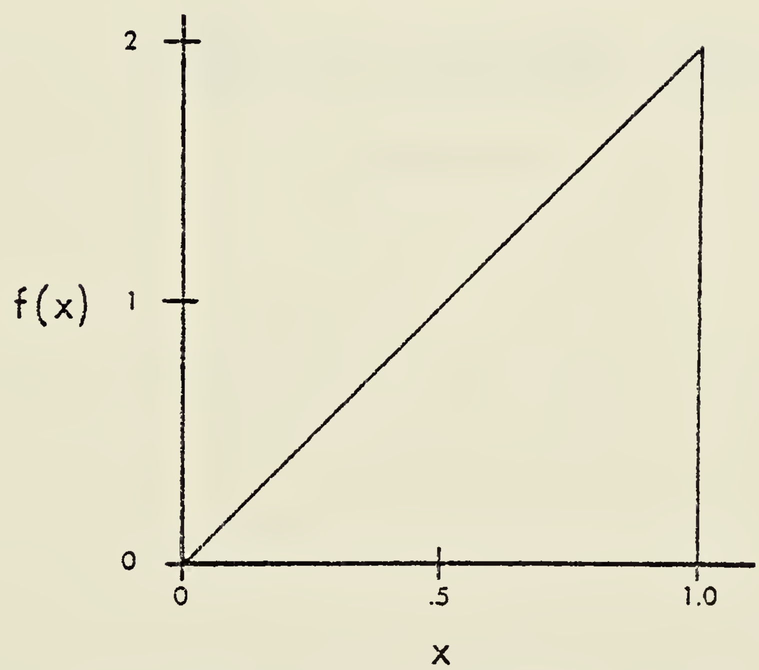


FIG. 4 DISTRIBUTION OF SHOVELS BY DISTANCE

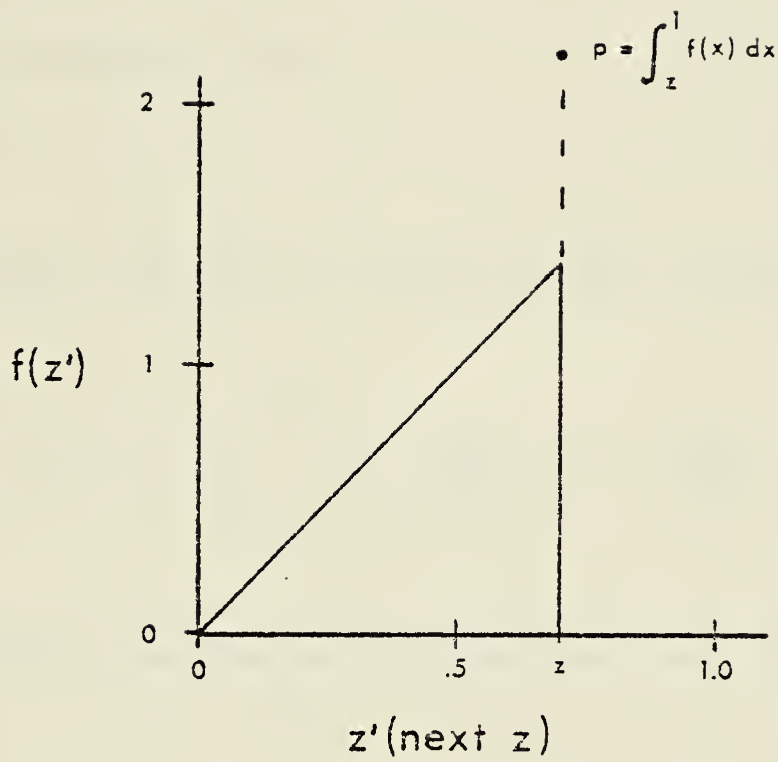


FIG. 5 TRANSITION PROBABILITIES FOR z

a higher cost one by twiddling our thumbs).

We can calculate the expected cost of such a decision strategy. For any T, let the probability of moving to $z \le T$ on any step be

$$P_T = \int_0^T f(x)\,dx = T^2 .$$

So, the expected number of steps before stopping is

$$N_T = \sum_{i=1}^{\infty} iP_T(1-P_T)^{i-1} = \frac{1}{P_T} = \frac{1}{T^2}$$

Also, we need the expected distance of the successful shovel

$$E[z|z\le T] = \frac{\int_0^T zf(z)\,dz}{P_T} = \frac{2T^3}{3T^2} = \frac{2T}{3}$$

So the expected cost is:

$$C_T = N_T + aE[z|z\le T]$$

$$= \frac{1}{T^2} + \frac{2aT}{3}$$

where locating a shovel has unit cost, and a shovel at unit distance has a cost of a. Now, the optimal T will be that T' for which C_T is minimal.

$$\frac{d}{dT} C_T = \frac{-2}{T^3} + \frac{2a}{3}$$

$$T' = (3/a)^{1/3}$$

For example, assume we can obtain a shovel from the outskirts of the city in the time it takes to locate 30 shovels-for-sale. Then $a=30$, and $T' = \sqrt[3]{(3/30)} = .464$. That is, if the edge of the city is 10 miles away, we should keep locating shovels until we find one within 4.64 miles.

Let us look at this problem in a different light. Suppose we adopt a "myopic" decision strategy, where to evaluate the cost of continuing, we assume we will locate only one more shovel; then choose this shovel or the nearest one found previously. As a decision tree:

```
                      ____
A ___________| OR |___________ B
|            |____|            |
|                              |
SVL1                     seek a nearer shovel
```

Clearly, the cost of decision A is $C_A = az_1$. To evaluate C_B, we consider only

```
                    | B
                   _|__
D ___________| OR |___________ E
|            |____|            |
|                              |
SVL2                         SVL1
```

here clearly $C_E = C_A = az_1$. We choose alternative D

iff $C_D \le C_E$, so to evaluate C_B we need $P[C_D \le C_E]$ and $E[C_D \mid C_D \le C_E]$. These are precisely the P_T and $E[z \mid z \le T]$ from above, with $T = z_1$. That is,

$$P[C_D \le C_E] = z_1^2 \quad \text{and}$$

$$E[C_D \mid C_D \le C_E] = 2az_1/3$$

So, C_B = cost of locating SVL2 + (probability of choosing D) * (expected cost of SVL2, given choice D) + (probability of choosing E) * (cost of SVL1). Substituting,

$$C_B = 1 + a(z_1 - z_1^3/3)$$

Now, let us determine under what conditions decision B should be made, that is, for what z_1 is $C_B < C_A$

$$1 + a(z_1 - z_1^3/3) < az_1$$

$$z_1 > (3/a)^{1/3}$$

That is, we will stop seeking shovels when we have located one which is as close as $(3/a)^{1/3}$; <u>note this is precisely the same as the stopping threshold T' obtained by a full analysis. However, it is obtained by looking only one step ahead,</u> instead of considering all the possible outcomes of an infinite number of steps. Apparently, breaking the infernal loop mentioned at the

outset after only one iteration need not have disastrous results.

Such serendipity is by no means guaranteed for all problems, of course. However, looking ahead one step is always better than not looking ahead at all. I propose that a suitable structuring of problems makes such a myopic strategy adequate: that structure commonly called hierarchal planning. In fact, making local decisions compatible with global goals can be viewed as the entire objective of a planning system.

3.2 Complications of Reconsideration

Choosing among alternatives is the subject of classical decision theory. A central result is that each alternative may be evaluated by a single-valued utility [jj] representing the expected gain resulting from its selection. For example, in Feldman's formulation of the monkey and bananas problem, suppose a box could be heavy or light, heavy boxes costing 12 and light ones 8. Suppose P[heavy] = 1/2. Then the alternative of using a box would have expected cost (negative utility) of (.5*12)+(.5*8) = 10. This is represented graphically in Figure 6a , where the round node indicates an externally determined choice (as opposed to a decision, which will be a square node).

Suppose the monkey has an alternative, say of asking a nearby man "Hey Mister, can you fetch me the bananas?"

The man may respond "Sure, monkey" (probability 1/2) or "Yeh, but it'll cost you 40, you hairy ape" (Figure 6b).

Now let us suppose there is some planning cost, or other front-end load, for each alternative, say 6 for the box (the cost of walking to it and giving it a trial push) and 2 for the man (cost of talking). Using a square node to represent the decision, we have the tree of Figure 6c. Usual methods let us calculate the expected cost of the man as 2+20 = 22, and of the box as 6+10 = 16. An optimal decision then would be to use the box, and the decision node would acquire this cost (as shown).

However, this naive application of Feldman's proposal ignores the possibility of reconsidering a choice (backing up), something any successful system must be able to do. For example, if the man says "That'll be 40" the monkey need not spend the 40 but may reply "Get real, man, I'll use the box".

This introduces a second level of decision making. Similarly, if he considers the alternative at the second level, he is faced with a third level (trivial) decision as to which to use. Our decision tree has grown considerably to include these refinements (Figure 7a). We now see that considering the box has expected cost of 6+7 = 13, not 16 as before; the cost of the man drops even more dramatically, from 22 to 2+8 = 10. Most significantly, we now see that we expect to spend fully 3

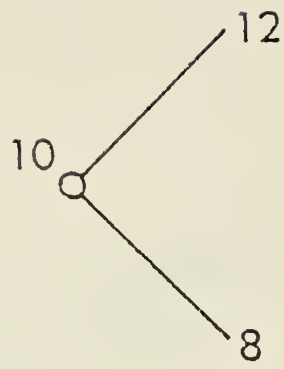


FIG. 6a UTILITY OF BOX

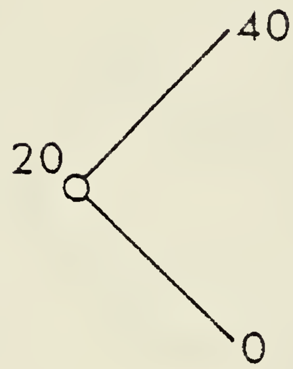


FIG. 6b UTILITY OF MAN

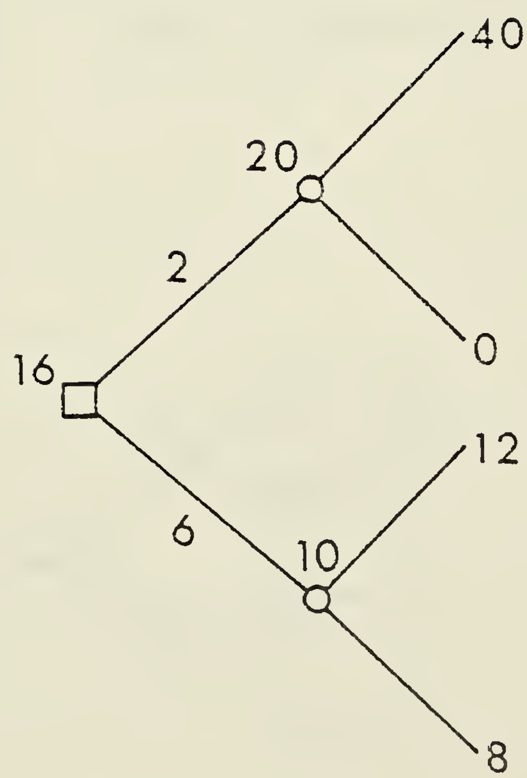


FIG. 6c NAIVE DECISION TREE

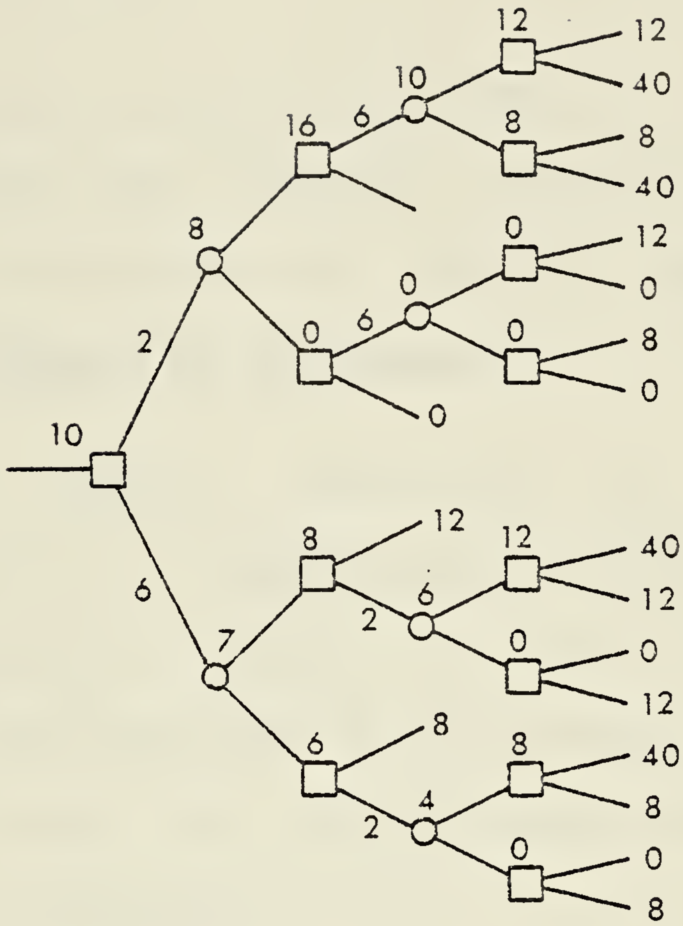


FIG. 7a FULL TREE WITH BACK UP

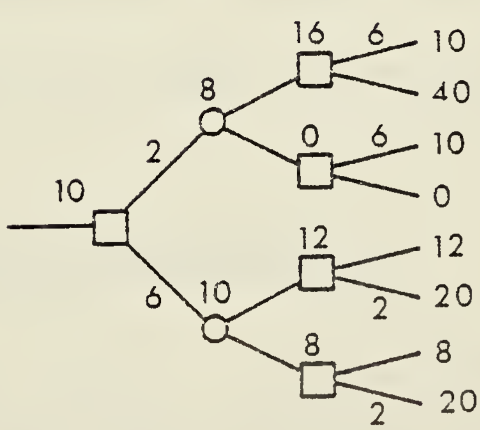


FIG. 7b ONE LEVEL TREE

units less by considering the man first, the opposite decision from before.

We note that to each alternative one can still attach a single-valued cost (utility), with the optimal decision still being the minimum cost. However, <u>the cost of each alternative depends on the other</u>.

Clearly, though, elaboration of the full tree of possibilities is a computational nightmare. Since the decision theoretic optimal decision is based on such elaboration, the strategy of doing such a thing in the real world (that is, including the cost of "thinking") could never be optimal.

My proposal then is to limit this elaboration to one level. That is, the second level decisions would be handled in the "naive" manner, excluding further reconsideration. The tree in this instance becomes that of Figure 7b. Relative to the full tree, we still make the optimal decision and know its correct cost.

3.3 Generalized representation

What sort of a generalized representation of alternatives does this lead to? The distribution of costs after an "external choice" (in general, any event making available new information) can be continuous rather than discrete-valued as considered above. As well as the cost,

our estimate of probability of success of the alternative may be revised.

3.3.1 Need for probability estimates

Why not, as Feldman [3] and others have done, ignore the probability of success of an alternative, and plan only using its effect on utility?

There are two possible formulations of a utility measure: as a positive utility to be maximized, or as a cost to be minimized. While in theory one is simply the additive inverse of the other (± a constant), as concepts they lead to distinguishable paradigms, the offset constant becoming infinite.

It is natural with the utility paradigm to let u, the utility of failure, be 0. Feldman arbitrarily assigns 200 units of positive utility to achieving the goal of eating the bananas, and an unspecified utility to not eating (failure). Suppose now the monkey has available a plan for feeding costing 200 units, with probability of success 1/2. Clearly the (expected) utility of the plan is $-200+(.5*200)+(.5u) = .5u-100$. Now for $u > -200$, $u > .5u-100$, that is, the utility of failure is greater than that of the plan for feeding. So, the monkey will prefer to sit and starve to death rather than trying to eat.

It is intuitive to demand of any decision maker that <u>any</u> attempt to eat, no matter how costly or unlikely to succeed, is preferable to failure (starvation). This is because there is no state of suspended animation; failure incurs a cost that increases with time unboundedly. Since we are forced to talk of negative utilities, it is more natural to adopt the cost paradigm.

With costs, it is intuitive to assign a "cost" of 0 to success. Then let f be the cost of failure. What is the expected cost of an alternative not guaranteed to succeed? Clearly it is $t+(1-p)f$ where t is the expected cost to complete the alternative and reach a decision of success or failure (probability of success p). Now our intuitive demand that $t+(1-p)f < f$ for any permissible values of t and p implies $t < pf$ or $f > t/p$. So $f \longrightarrow \infty$ as $p \longrightarrow 0$ for fixed t. We cannot simply let $f = \infty$ however; the expected cost of any alternative not guaranteed to succeed would also be infinite. Usual decision theory demands that costs (utilities) be finite:

> . . . in an ordinary problem there is neither 'heaven' nor 'hell' in R [the set of rewards]. [2,p. 103]

But failure is hell. This assumes the failure is a "top level" failure, that is, one to which there is no alternative having any possibility of success.

What can we do? In practice, we only want to chose between "non-failure" alternatives, say $(A_1 \text{ OR } A_2)$;

failure can never be considered as an alternative. What is the expected cost E_1 of A_1? Clearly,

$$E_1 = t_1 + (1-p_1)t_2 + (1-p_1)(1-p_2)f$$

where A_1 and A_2 have cost and probability (t_1, p_1) and (t_2, p_2), respectively. Similarly,

$$E_2 = t_2 + (1-p_2)t_1 + (1-p_1)(1-p_2)f$$

and we chose E_1 if $E_1 \leq E_2$, that is, if

$$t_1 + (1-p_1)t_2 + (1-p_1)(1-p_2)f \leq t_2 + (1-p_2)t_1 + (1-p_1)(1-p_2)f$$

$$t_1 - (1-p_2)t_1 \leq t_2 - (1-p_1)t_2$$

$$\frac{t_1}{p_1} \leq \frac{t_2}{p_2}$$

Fortunately, the f terms cancel. That is, we need not be concerned with the cost of ultimate, top level failure in making decisions based on expected costs. It is useful to talk of the cost of an alternative <u>excluding the possibility of failure</u>. In the above example, E_1 would become $t_1 + (1-p_1)t_2$, and similarly for E_2. Note that the "internal" costs t_1 and t_2 are also of this type. All costs throughout the remainder of this thesis will exclude the possibility of failure, and will simply be referred to as "cost". This creates no problem when choosing among alternatives; however we note that failure cannot be

handled in a uniform way, as merely another alternative (but with infinite cost). We note now that cost depends on the characteristics of the alternative A_2; we cannot evaluate the cost (utility) of an alternative without knowing the other alternatives available. Therefore, we must characterize an alternative by both its internal expected cost and its probability of success.

3.3.2 Characterization of an alternative

Now that the need for a two component characterization of an alternative has been demonstrated, how shall an alternative be characterized as a two step process?

The first step will have an expected cost, which we term s. After this step, the (t,p) estimates are revised to (t',p'); this can be described by a bivariate distribuition function f(t',p'). The (prior) expected cost of the second step will be E[t'], so then the total expected cost must equal s+E[t']. Also, the process is assumed not to succeed during the first step (without loss of generality, since success is equivalent to a second step cost of 0), so p = E[p'].

For example, suppose in the monkey and bananas problem that the cost of pushing boxes is proportional to weight, which is normally distributed and can be determined by giving a box a trial push. Then if it costs

6 to walk to a (particular) box and give a trial push, f(t') will be normal. Suppose also that touching a box revises our estimate of its climbability; then p will also be revised (say normal also, independent of t). Then the alternative of continuing with this box will be characterized by

$$s = 6 \qquad f(t',p'): \text{ bivariate normal}$$

3.3.3 Choosing among alternatives

How are we to choose between alternatives so characterized? Our one level look-ahead criterion presumes simple decisions one level ahead; these are simply those derived in Section 3.3.1, namely to choose A_1 if

$$\frac{t_1}{p_1} \leq \frac{t_2}{p_2}$$

That is, the ratio t/p is a measure of the (un)worthiness of an alternative. It gains intuitive meaning from realizing that it is the expected cost to success of a series of alternatives of (t,p) pursued sequentially. For example, suppose we have an infinite set of boxes, each having probability p of containing a ball. If it costs t to search each box, the expected cost of finding a ball will be t/p. This use of t/p as a measure of worth has

been derived by others in this sequential search context, most recently by Simon and Kadane [11].

Now we can consider one level look-ahead choice. We consider the cost of choosing an alternative, say A_1, and investing the effort s_1 to revise our estimates of p_1 and t_1 to p'_1 and t'_1 according to distribution $f_1(t'_1, p'_1)$. Then we will use the simple criterion

$$\frac{t'_1}{p'_1} \le \frac{t_2}{p_2}$$

to choose whether to continue with A_1 or switch to A_2. Let U be the event that A_1 is so chosen; this will correspond to the shaded region (Figure 8) of the (t',p') plane bounded by the line $(t'_1/p'_1) = (t_2/p_2)$. Then the expected cost E_1 of choosing A_1 at the outset is

$$E_1 = s_1 + P[U]\{E[t'_1|U] + (1-E[p'_1|U])t_2\}$$
$$+ (1-P[U])\{t_2 + (1-p_2)E[t'_1|\text{not } U]\}$$

Similarly, if we let V be the event that $(t'_2/p'_2) \le (t_1/p_1)$,

$$E_2 = s_2 + P[V]\{E[t'_2|V] + (1-E[p'_2|V])t_1\}$$
$$+ (1-P[V])\{t_1 + (1-p_1)E[t'_2|\text{not } V]\}$$

Our decision criterion is now to chose A_1 if $E_1 \le E_2$.

This defines a binary comparison of ORed alternatives. Although this comparison is not obviously transitive (exceptions are difficult to imagine), it can be extended to chose one of n alternatives, as follows.

Consider a set of n alternatives $\{A_i: \quad 1 \le i \le n\}$. Numerically ranking the values (t_i/p_i) will produce a permutation r of the integers 1 to n, such that

$$\frac{t_{r(i)}}{p_{r(i)}} \le \frac{t_{r(i+1)}}{p_{r(i+1)}}$$

Then we can define probability of success

$$p = 1 - \prod_{1 \le j \le n} (1 - p_j)$$

and expected cost

$$E = \sum_{1 \le k \le n} t_{r(k)} \prod_{1 \le j \le k} (1 - p_{r(j)})$$

of trying the alternatives in the order defined by r. To develop a one level look-ahead choice, we consider the cost E_i of starting with A_i and then either completing A_i or switching to try the remaining alternatives in the order defined by r. Define q_i to be the probability of failure of all alternatives except A_i:

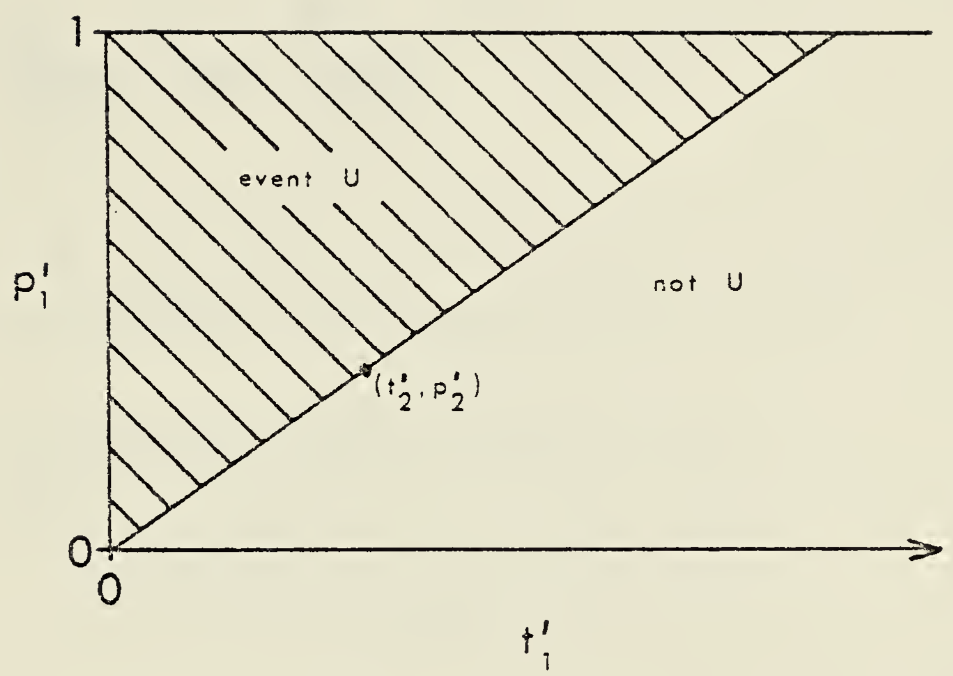


FIG. 8 DECISION EVENTS

$$q_i = \prod_{\substack{1 \le j \le n \\ j \ne i}} (1-p_j)$$

and C_i to be the expected cost of trying the alternatives except A_i:

$$C_i = \sum_{\substack{1 \le k \le n \\ r(k) \ne i}} t_{r(k)} \prod_{\substack{1 \le j \le k \\ r(j) \ne i}} (1-p_{r(j)})$$

Then let I be the event that $(t'_i/p'_i) \le (C_i/(1-q_i))$, that is, that we choose to continue with A_i. So

$$E_i = s_i + P[I]\{E[t'_i|I] + (1-E[p'_i|I])C_i\}$$
$$+ (1-P[I])\{C_i + q_i E[t'_i|\text{not } I]\}$$

Because of shared terms among the C_i, the complexity of this computation increases only linearly with n.

The question now arises: if costs of altervatives are interdependent and so can't be used alone, why not characterize alternatives by only their unworthiness t/p? This is clearly independent of the alternative, yet would allow simple and look-ahead decisions analogous to those considered above, with only one parameter instead of two. The answer lies in the other uses of characterizations (other than choosing among ORed alternatives). For example, it is clear how to combine the (t,p) characterizations of conjuncts to get a characterization

of the conjunction, but this is not so with t/p.

3.4 Practical representation

A problem of practical importance now arises. What sort of a computationally manageable approximation of the bivariate distribution function $f(t',p')$ is best? Four categories of representation were considered.

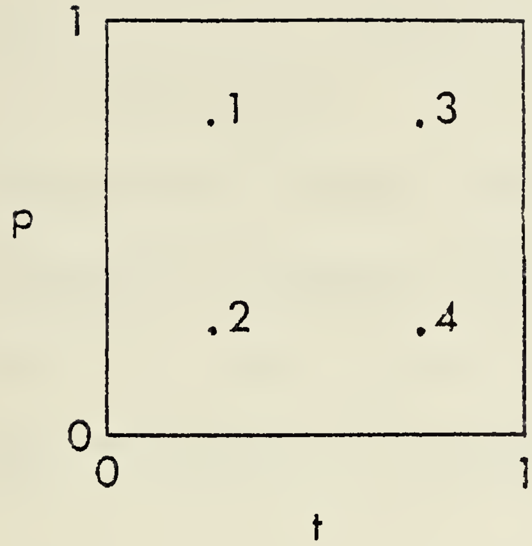
3.4.1 Parameterized distribution

This category was dismissed because of the difficulty of evaluating the $E[t' \mid (t'/p')<w]$ and $E[p' \mid (t'/p')<w]$ terms of the decision criterion. For example, although polynomial approximations are available for the bivariate normal, the shape of the regions $(t'/p')<w$ is intractable.

3.4.2 Discrete points

This type of representation approximates $f(t',p')$ by a number of points (t_i,p_i) each with probability q_i. For example, a uniform distribution on $t<1$ might be approximated as in Figure 9.

This was rejected as being too pessimistic in cases of small t/p; in the example, the approximation indicates no possibility of t/p going lower than 1/2. It is often, of course, just such "remote" possibilities that we are interested in.



i	q	p	t
1	.25	.75	.25
2	.25	.25	.25
3	.25	.75	.75
4	.25	.25	.75

FIG. 9 DISCRETE REPRESENTATION

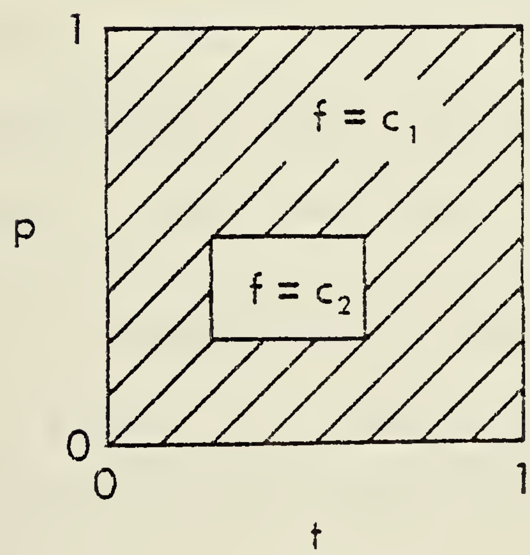


FIG. 10 STEP FUNCTION REPRESENTATION

3.4.3 Rectangular prisms

This representation would approximate the value of f(t',p') as a constant in a region bounded inside and out by rectangles, as in Figure 10. Thus we have a two dimensional step function; two to five steps would probably be needed. The problem here too lies in computing E[t'|U] and E[p'|U] terms: the number of cases resulting from dividing a rectangle with a line is excessive.

3.4.4 Set of lines and points

The representation finally adopted is of this type. All probability is condensed onto a set of lines and points. The one dimensional distributions along the lines are represented as a piecewise linear function with up to five pieces. This type of representation should be less crude than either the discrete only or step functions considered above, as well as being computationally simpler than a step function.

Several sorts of line condensations were considered. A number of variations had lines radiating from some central point, usually the mean (Figure 11).

A common problem of radial lines is that different points will represent the condensation of different areas. For example, the representation of a uniform distribution on t'<1 might be as in Figure 12, where the distribution

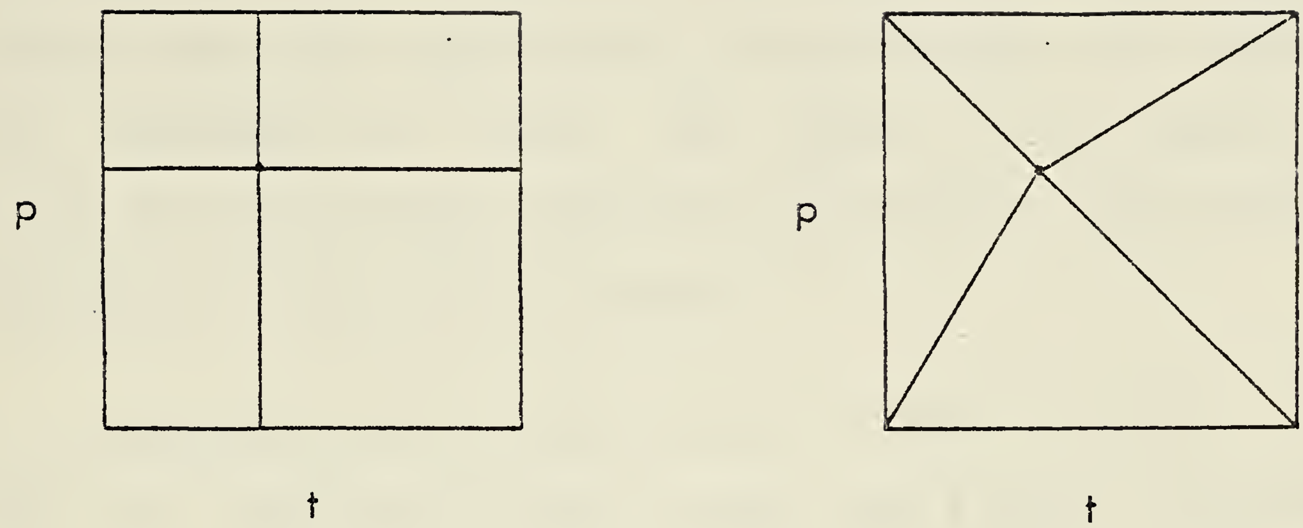


FIG. 11 POSSIBLE RADIAL REPRESENTATIONS

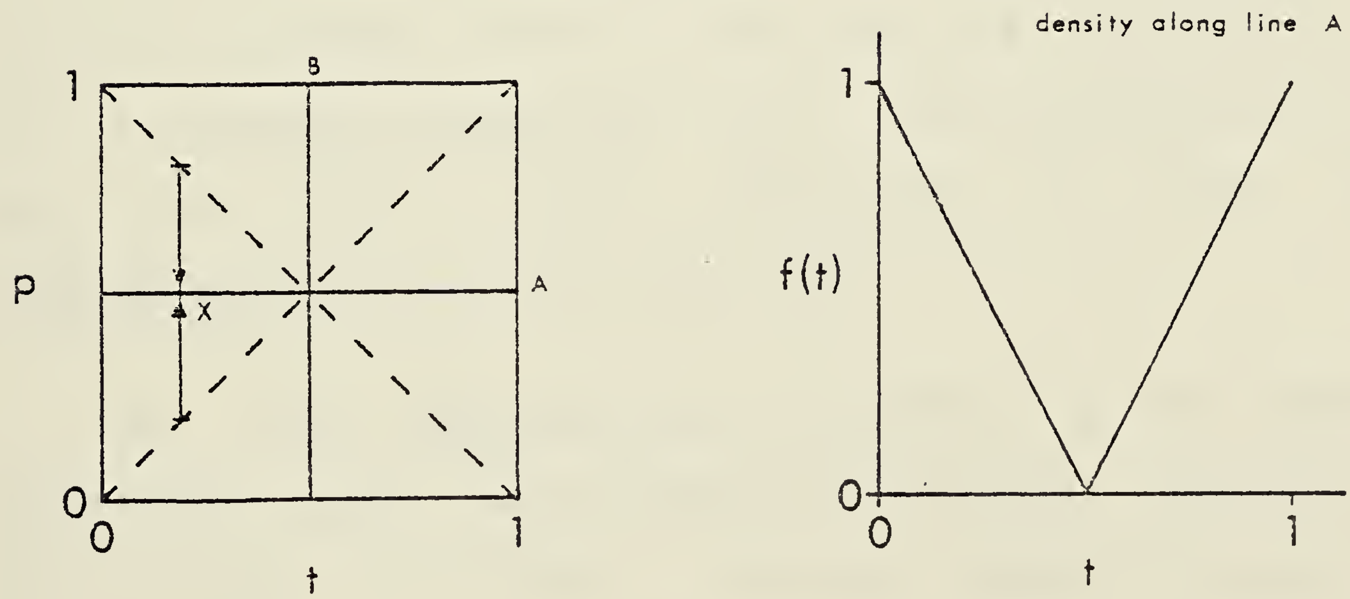


FIG. 12 RADIAL REPRESENTATION OF UNIFORM

along line A is defined by condensing into points X on the line the probability between the diagonals. The source of the difficulty is computing probabilities for intervals on these lines: the distribution function must be scaled by the shrinkage of area; the result is in general (for reasonable shrinkage functions) quadratic. Lines should be parallel to avoid this problem.

The question of slope of the lines remains. Should they be parallel to the t axis, the p axis, or somewhere between? The preferred representation has them parallel to the t axis; this choice was motivated by the non-linear distortion of lines of any other slope on combining representations at an OR node using a combining method later judged nonpreferrable.

However, I believe the choice is still sensible. Lines not parallel to one of the axes are unappealing and of no apparent advantage. And it would seem desirable to have more detail about variations in cost than probability, rather than vice versa.

So, the representation proposed is to condense probability onto a set of five or less lines parallel to the t axis, and into a similar number of points (if desirable). The selection of numbers and locations of the lines and points, and condensation rules, are left to ad hoc decision.

3.5 Backing up characterizations

The control structure I am proposing is like that of MULTIPLE. This requires the ability to "back up" characterizations over OR and AND nodes, so that the characterizations of higher nodes may be refined, and possibly choices altered, as elaboration proceeds.

3.5.1 OR nodes

Let us treat the OR node first. Here we have two alternatives A_1 and A_2 (n-ary ORs can be strung out as a sequence of binaries), one of which will be the "choice" by the proposed decision criterion, say A_1.

The simplest possibility is for the OR to adopt the characterization of the choice (A_1). However, this says nothing of the advantage that is gained by having a non-failure alternative: the characterization is the same if A_2 is only slightly less good or if A_2 is total failure. The error in the backed up probability of success is especially glaring.

A way of combining the characterizations is suggested by the formula for expected cost of starting with A_1:

$$E_1 = s_1 + P[U]\{E[t'_1 \mid U] + (1 - E[p'_1 \mid U])t_2\}$$
$$+ P[\text{not } U]\{t_2 + (1 - p_2)E[t'_1 \mid \text{not } U]\}$$

Now consider $t'(t'_1, p'_2)$, the expected remaining cost of the OR given that the initial step of A_1 has been taken (at a cost of s_1), and we have values for t'_1 and p'_1. If $(t'_1/p'_1) \leq (t_2/p_2)$ (event U; see Figure 8 above), we continue with A_1 and $t' = t'_1 + (1-p'_1)t_2$. If not, we switch to A_2 and $t' = t_2 + (1-p_2)t'_2$. That is,

$$t'(t'_1, p'_1) = \begin{cases} t'_1 + (1-p'_1)t_2 & \text{if } \dfrac{t'_1}{p'_1} \leq \dfrac{t_2}{p_2} \\[2ex] t_2 + (1-p_2)t'_1 & \text{if } \dfrac{t'_1}{p'_1} > \dfrac{t_2}{p_2} \end{cases}$$

Similarly, the probability of the OR:

$$p'(t'_1, p'_1) = 1-(1-p'_1)(1-p_2)$$

These functions will map the $f_1(t'_1, p'_1)$ distribution into a new distribution for the OR. The initial stage cost s of the OR would be s_1; t and p of the OR would be the expected values of the transformed distribution:

$$t = s_1 + E[t']$$

$$p = E[p']$$

To evaluate this method of combination, it is useful to consider an example. Suppose for A_1 that $f_1(t'_1, p'_1)$ is uniform on $t'_1 \leq 1$ (and zero for $t'_1 > 1$). Let s_1 be vanishingly small, and let $t_2 = 1/2$, $p_2 = 1/2$. Then $t_1 =$

1/2 and $p_1 = 1/2$. Suppose we condense the uniform f_1 onto five lines at p = .1,.3,.5,.7, and .9 (see Figure 13a). The transformed f(t',p') is in Figure 13b. While the expected probability of success rises from .5 to .75, the expected cost also rises from .5 to .6675, so the unworthiness t/p falls from 1 to .89.

However, this improvement in gross parameters is achieved at the expense of the possibility of large improvement. While with A_1 the cost and unworthiness could fall to zero, the OR node indicates that these will not fall below .05 and .0556 respectively. Thus while our decision criterion may well prefer A_1 to a potential A_3 with $t/p < .0556$, (A_1 OR A_2) will never be preferred. This does not seem acceptable. Why should supplying a non-failure alternative make A_1 less desirable?

The answer lies in the status of failure as a non-alternative (see Section 3.3.1): the characterization of A_1 is not what one would get by combining (A_1 OR Failure). Indeed, the difficulty with characterizing such a combination is the reason for making failure an exception.

We see the problem at another level. The transformation proposed for (A_1 OR A_2) was derived on the assumption that the alternative to (A_1 OR A_2) was failure; this will not usually be the case: the weakness of this transformation appears when we consider a third

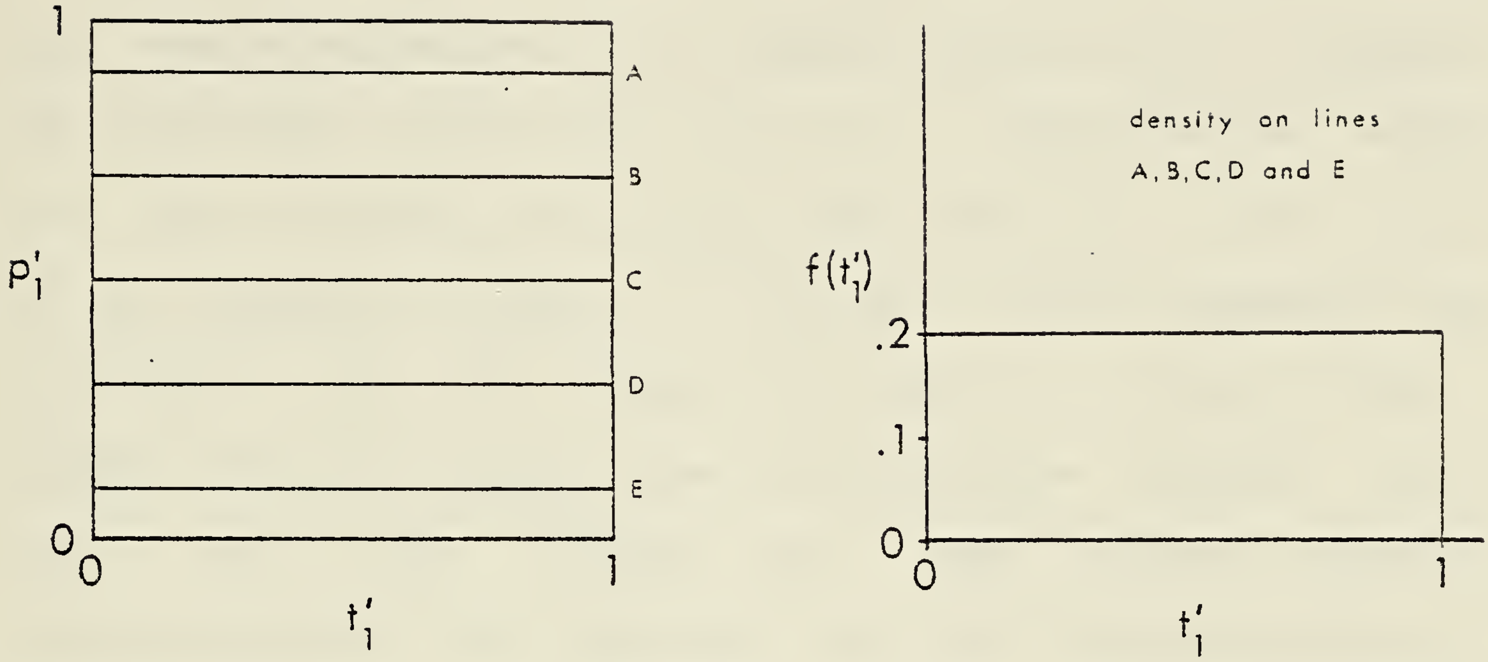


FIG. 13a PARALLEL REPRESENTATION OF UNIFORM

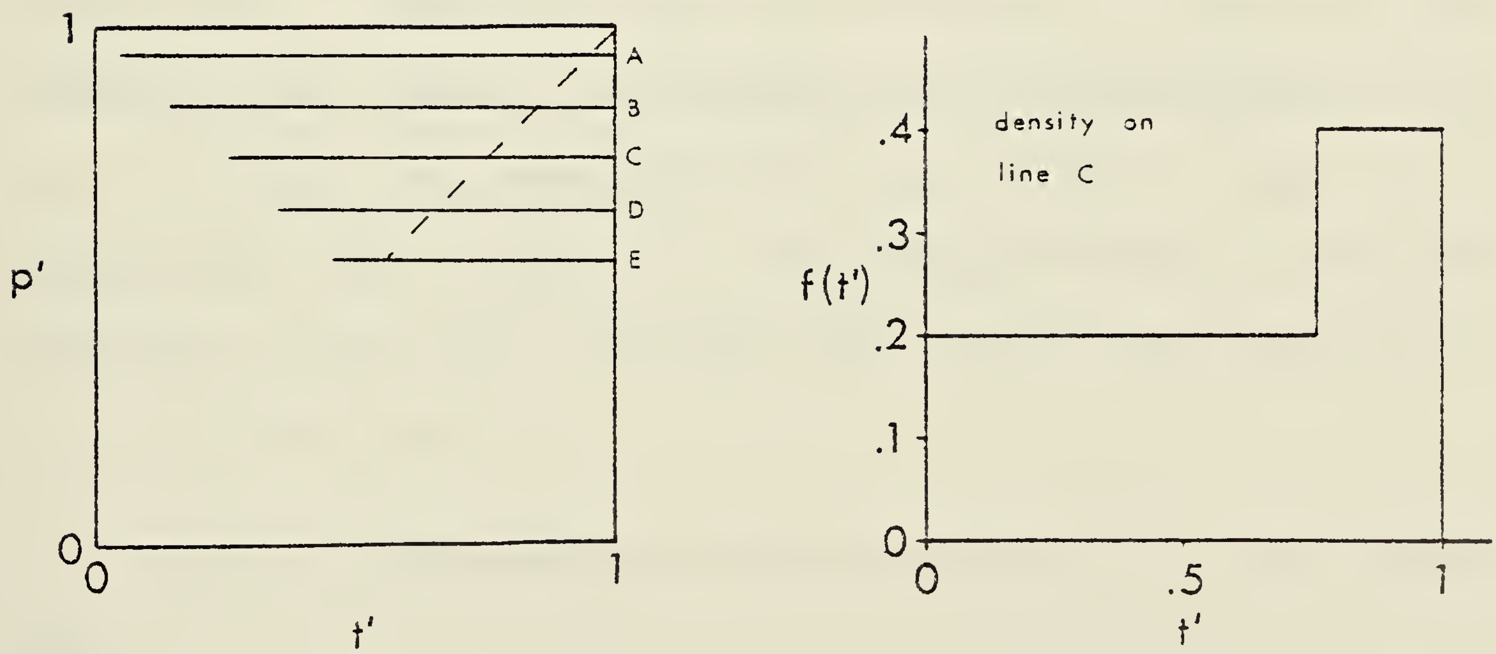


FIG. 13b ORed UNIFORM, NOT RECOMMENDED

non-failure alternative.

What can we do? Rather than go back to simply using the characterization of the choice alternative, I suggest the following. Divide the $f_1(t'_1, p'_1)$ distribution along the line $(t'_1/p'_1) = (t_2/p_2)$. Let $f(t',p') = f_1(t'_1, p'_1)$ in the region $(t'_1/p'_1) \leq (t_2/p_2)$, and condense the probability in the region $(t'_1/p'_1) > (t_2/p_2)$ into a single point at (t_2, p_2). The illustrative example would then look like Figure 14. This allows improved alternatives to be reflected in the characterization, without sacrificing the desirability of the disjunction in further comparisons.

3.5.2 AND nodes

Let us now consider backing up over an AND node (G_1 AND G_2 AND ... AND G_n), where the conjoined subgoals are characterized like the alternatives dicussed above, that is, G_i will be characterized by initial cost s_i, probability p_i, cost t_i, and a distribution of revised estimates $f_i(t'_i, p'_i)$. The AND will be characterized be s, t, p, and $f(t',p')$.

Assuming independence of the subgoals, it is clear that

$$p = \prod_{1 \leq i \leq n} p_i \quad \text{and} \quad t = \sum_{1 \leq i \leq n} t_i .$$

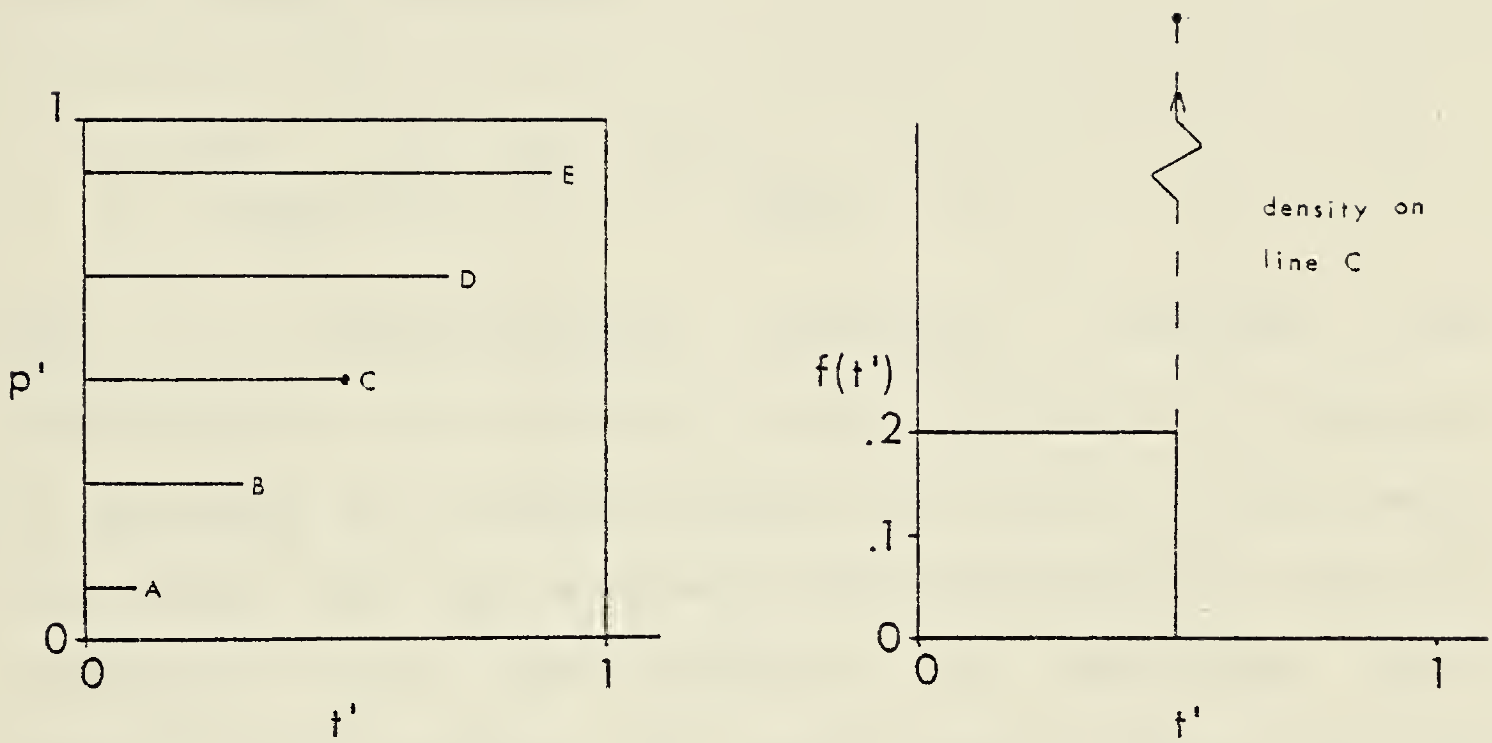


FIG. 14 ORed UNIFORM, RECOMMENDED

If we assume the subgoals are pursued in parallel (or rather, breadth-first), then we would have

$$s = \sum_{1 \leq i \leq n} s_i$$

and f(t',p') would be the distribution of the sum and product random variables

$$p' = \prod_{1 \leq i \leq n} p'_i \quad \text{and} \quad t' = \sum_{1 \leq i \leq n} t'_i \; .$$

This is computationally difficult, involving the convolution of distributions. Also, pursuing the subgoals in parallel is a poor strategy; we should do first what will affect our (t,p) estimates most drastically, that is, "schedule" first (for planning, not execution) that subgoal which effects the most rapid change in the character of the problem.

Consider the problem of scheduling the planning of two conjoined subgoals G_1 and G_2. The argument is made in relation to an alternative A_4:

```
 A                         ____
  3  ___________| OR |__________
    |             |____|          |
    |                             |
   _____                          A
 __| AND |__                       4
|  |_____|  |
|           |
G           G
 1           2
```

We will consider the expected cost of $((G_1 \text{ THEN } G_2) \text{ OR } A_4)$

against $((G_2 \text{ THEN } G_1) \text{ OR } A_4)$. Suppose investing s_1 in G_1 does not affect our estimates of p_1 and t_1, that is, $p'_1 = p_1$ and $t'_1 = t_1 - s_1$. Then if we choose A_3 initially, we will still choose A_3 after s_1, and (presumably) invest s_2 in G_2. Suppose after s_2 that $(t'_2=0, p'_2=0)$ and $(t'_2=1.0, p'_2=1.0)$ each occur with probability 1/2. Then after investing s_2 we can choose between A_3 and A_4. Now, the expected cost E_{21} of $((G_2 \text{ THEN } G_1) \text{ OR } A_4)$ is

$$E_{21} = s_2 + .5t_1 + .5t_4$$

and similarly

$$\begin{aligned} E_{12} &= s_1 + s_2 + .5t'_1 + .5t_4 \\ &= s_1 + s_2 + .5t_1 - .5s_1 + .5t_4 \\ &= E_{21} + .5s_1 \end{aligned}$$

that is, when doing $(G_1 \text{ THEN } G_2)$, half the time planning step s_1 is wasted.

This motivates our intuition that when there is an alternative to a goal, we should pursue the goal in such a way that information causing us to switch to the alternative should be utilized as soon as possible. Further analytical justification is in the comparison of alternatives with normal $f(t')$ (Section 3.6); the larger the standard deviation of t', the more preferred the alternative to a uniform one.

How can we apply this to scheduling conjoined subgoals? I propose scheduling first that subgoal which maximizes the expected rate of change of the t/p ratio of the AND.

Let us construct a measure of this rate of change for (G_1 THEN G_2). Define event U that t/p decreases after investing s_1 in G_1, and event V = not U. Then a reasonable measure of change of t/p will be the expected absolute value of the change:

$$P[U](B_0 - B|U) + P[V](B|V - B_0)$$

$$\text{where} \quad B_0 = \frac{t_1 + t_2}{p_1 p_2}, \quad B|U = \frac{E[t'_1|U] + t_2}{E[p'_1|U]p_2}$$

$$\text{and} \quad B|V = \frac{E[t'_1|V] + t_2}{E[p'_1|U]p_2}$$

This is computationally easier if events U and V are independent of G . Define event U_1 that $(t'_1/p'_1) <$ (t_1/p_1), and V similarly; then the change measure will be

$$C_1 = P[U_1](B_0 - B|U_1) + P[V_1](B|V_1 - B_0)$$

Clearly, the $E[t'_1|U_1]$ etc. terms can be calculated for G_1 independently of G_2, so evaluating C_1 is a simple matter of multiplication and addition.

Rate of change of unworthiness then would be C_1/s_1; this can be defined similarly for any one of n subgoals. They would be scheduled for planning then by ordering the C_i/s_i ratios.

Let us return to the question of backing up the characterization of subgoals to an AND node, given the (planning) schedule (G_1 THEN G_2 THEN ... THEN G_n) as defined above. We can then let the initial step s of the AND be s_1, that is, the first step of G_1 is considered to be the first step of the conjunction. Then

$$t = \sum_{1 \le i \le n} t_i \quad ; \quad p = \prod_{1 \le i \le n} p_i$$

and $f(t',p')$ be $f\ (t',p')$ translated by

$$T = \sum_{2 \le i \le n} t_i$$

and scaled by

$$P = \prod_{2 \le i \le n} p_i$$

For example, if n=3 and all subgoals are characterized by the uniform distribution considered before (Figure 15), then $f(t',p')$ will be as in Figure 15.

This method of combination can be critized on the same grounds as the first proposed method for OR nodes (Section 3.5.1); possibility of small values of t (and large p) is unduly pessimistic (ie. 0). However, no easy

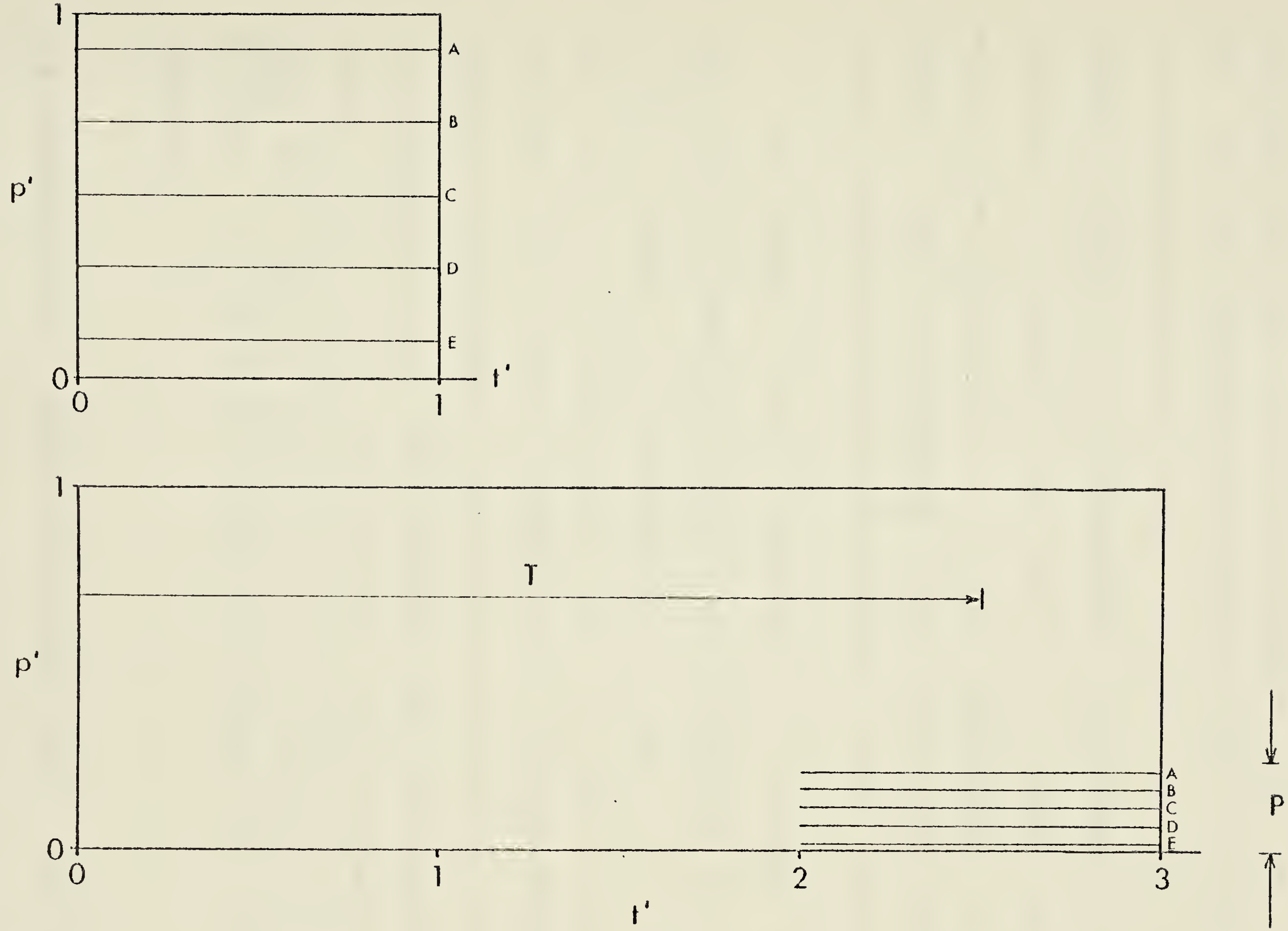


FIG. 15 ANDed UNIFORM

alternative is available.

Also, using the elaboration of a single subgoal to represent the "next step" of a conjunct may not be the best way of splitting the conjunct into two steps. That is, the first step of a single subgoal may have little overall effect on a conjunction of perhaps 20 similar subgoals -- it would be more natural to consider the first step of the conjunction to be the combination of first steps of <u>all</u> the subgoals in this case. Such refinement however seems too remote and difficult to deal with here.

The concept of using expected rate of change of the t/p ratio to schedule subgoals can be applied to OR nodes as well. However, in the disjunctive case, the expected cost of the OR is dependendent upon the order in which the disjuncts are tried, a complication not encountered with conjuncts. While not an impossible complication, there is no indication that this method will help achieve our ultimate goal, lowest expected cost. Lowest expected cost leads us to rate of change of t/p for conjuncts, but leads directly to the choice method of Section 3.5.1 above for disjuncts.

<u>3.6 Approximating a multi-step process with a two step one</u>

Suppose we have a multi-step process, after each step of which we acquire more reliable information on remaining cost. Where should we break such a process into only two

stages, for consideration by our one level look ahead decision criterion?

Consider two drills D_1 and D_2 attempting to reach a pool of oil (Figure 16). The oil is depth B_1 below D_1 and B_2 below D_2; define difference $d = B_1 - B_2$. For D_1, investment of unit effort deepens the hole y feet. For D_2, unit effort deepens the hole X feet, where X is a normal random variable with mean y and standard deviation σ_X; define $r = \sigma_X/y$. Each process can be split into two stages in infinitely many ways, characterized by the number s of units invested in the first stage. For D_2, expected depth drilled Z will be sy, with a variance of $s\sigma^2$, or $\sigma_Z = \sqrt{(s\sigma^2)} = ry\sqrt{s}$. Thus second stage cost $t'_2 = (B_2-Z)/y$, and $E[t'_2] = (B_2-sy)/y$; t'_2 has variance $s\sigma^2/y^2$.

The key observation is that for a uniform process of this type, the <u>variance</u> of the second stage cost is proportional to the cost of the first stage.

What size of first step will represent the use of two stages to maximum advantage? If the first step is too small, we gain little information about the possible improvement in overall cost, since the variance is also small. If the first step is too large, we already have such a large investment in the process that we lose the advantage of having an alternative available. It would seem there should be an optimum between the two extremes.

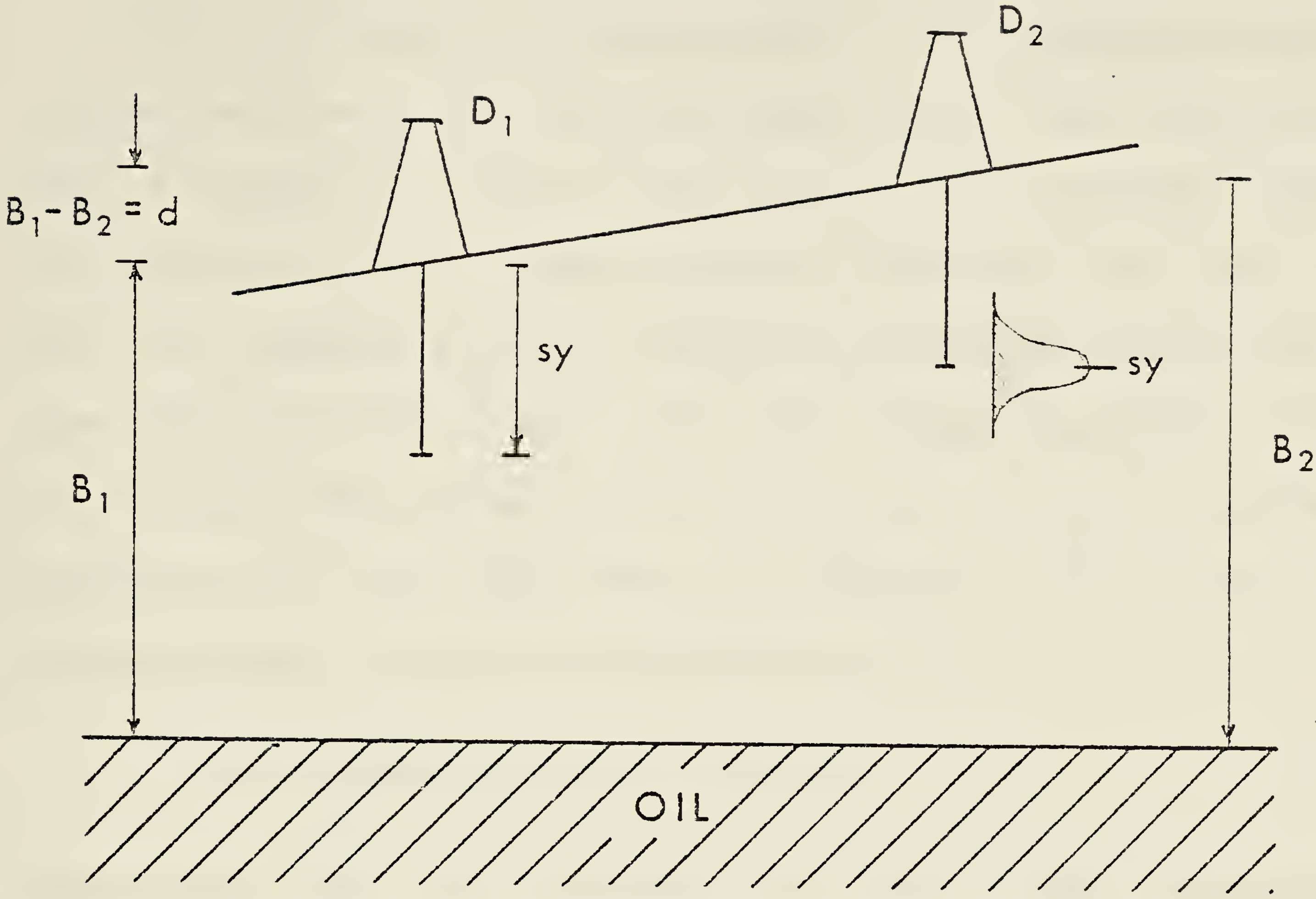


FIG. 16 OIL WELL EXAMPLE

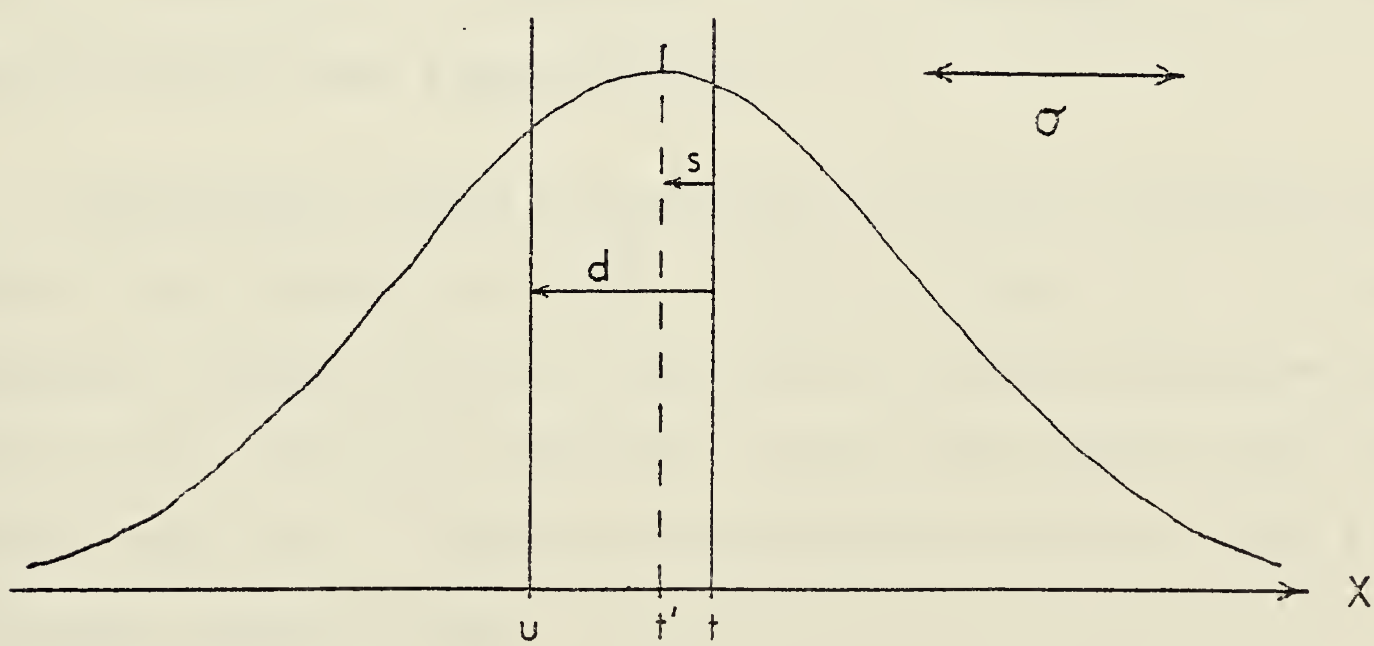


FIG. 17 TWO STAGE VERSUS ONE STAGE

Let us consider this problem in more general terms. Suppose we have a two stage process where the remaining cost X after a first stage of s is a normal random variable with mean t' and standard deviation σ, and we wish to compare it with a second process of fixed cost u (one stage process; Figure 17). The overall expected cost of the two step process will be $t = s+t'$. Define the cost difference $d = t-u$. The cost of beginning with the two stage process and then reconsidering will be

$$C = s + P[X<u]E[X|X<u] + P[X\geq u]u$$

Considering this as a function of u, let us seek solutions to the equation $C=u$. At this point, the expected cost of beginning with either process is the same, although the two stage process has independent expected cost $u+d$. That is, d can be considered as a measure of worth for the two stage process: it represents the advantage of the two stage over the one stage process.

Numerically derived solutions to $C=u$ are tabulated in Table 1 and plotted in Figure 18, in terms of d, for different values of first stage cost s (both normalized in units of σ). That is, the first table entry means that for first step cost s equal σ, the solution to $C=u$ is for $u = t-d$ with d equal 0.16.

What is the implication for a uniform process like the drilling of two wells, where σ^2 is proportional to s?

TABLE 1 SOLUTIONS TO C=u

$\frac{s}{\sigma}$	$\frac{d}{\sigma}$	$\frac{s}{\sigma}\ \frac{d}{\sigma}$
1.0	0.100	0.100
0.5	0.312	0.156
0.4	0.398	0.159
0.3	0.518	0.155
0.2	0.697	0.139
0.1	1.00	0.100
0.08	1.10	0.088
0.06	1.23	0.074
0.04	1.41	0.056
0.02	1.68	0.034
0.01	1.92	0.019

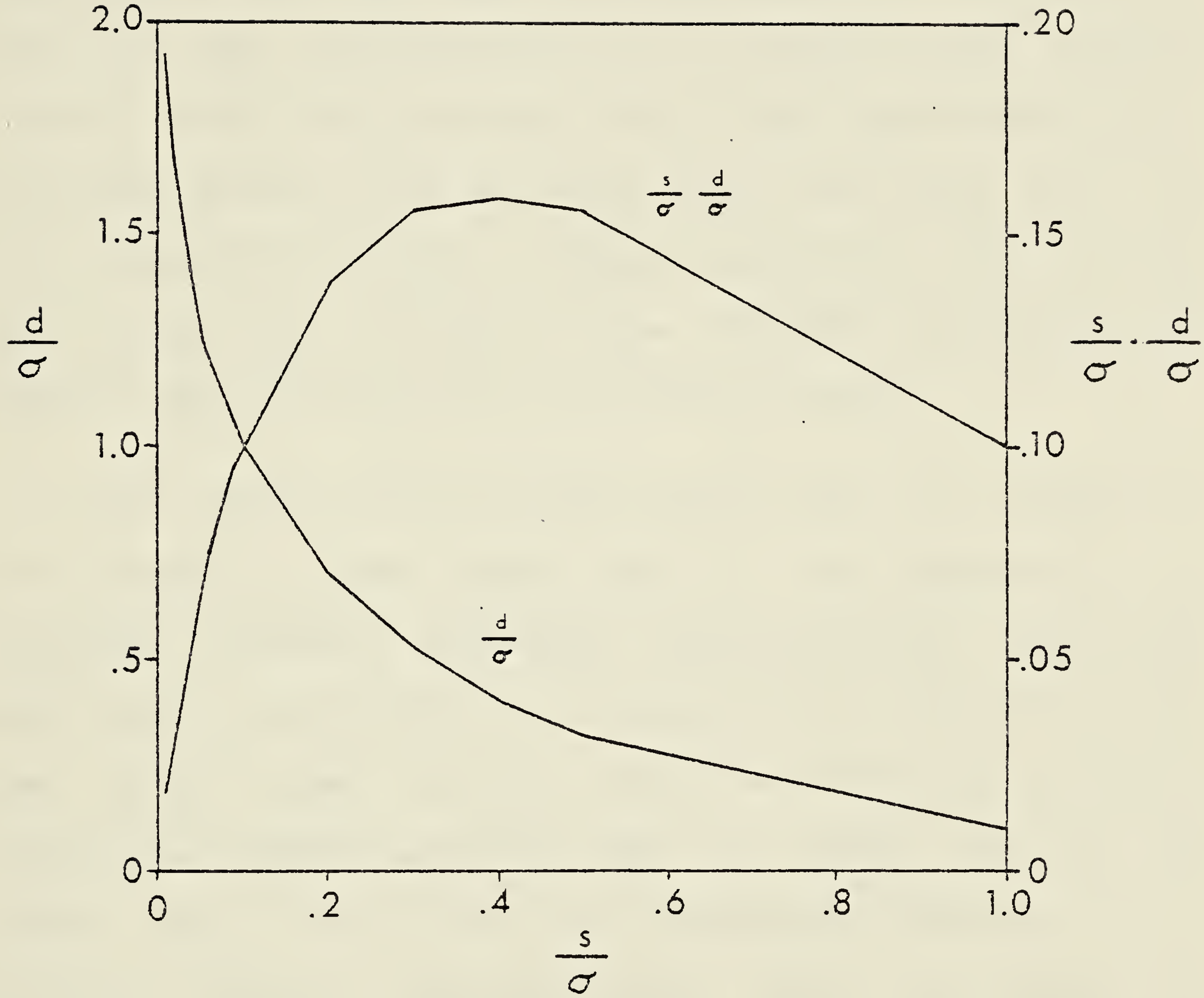


FIG. 18 PROPERTIES OF SOLUTIONS TO C=U

To represent the process to maximum advantage, we should chose the first step cost s giving maximum d. In this uniform case, for some constant b, $\sigma^2 = bs$ so $\sigma = bs/\sigma$, and $d = \sigma(d/\sigma) = b(s/\sigma)(d/\sigma)$. So, from the plot of $(s/\sigma)(d/\sigma)$ we can see that d is a maximum for $s = .4\sigma = d$. That is, the attractiveness of a multi-stage uniform process over a single stage one is best approximated by a two stage process with the first stage cost equal 4/10 of the standard deviation of second stage cost. That is, the first stage cost is chosen so the standard deviation of second stage cost will be 2.5 times as great.

What about the more usual cases where the process is not uniform? I have argued that it is reasonable to structure planning processes hierarchically, where the most drastic refinements to remaining cost estimates are done first. Then variance will be less than a linear function of effort invested, that is $\sigma^2 \leq bs$, and if we define $g(s) = \sigma^2/s$, then g is a decreasing function of s, and so $d = \sigma(d/\sigma) = g(s)(s/\sigma)(d/\sigma)$. Then the maximum d will correspond to smaller s than the maximum for g(s) a constant. That is, with hierarchical planning, we choose first stage cost so that the standard deviation of remaining cost is greater than 2.5 times first stage cost (as is optimal for the uniform case).

3.7 Non-termination

It is possible that a planning process may not terminate, that is, it may run on forever without either succeeding or failing. The expected cost of the process may then be infinite. Moreover, this may be true of a desirable process, for example, one which succeeds after unit effort with probability 0.99, but runs on forever otherwise. If we terminate this process after one unit, we convert it to one which succeeds with probability 0.99 and fails otherwise. Thus by establishing an upper cost limit, and regarding cases exceeding this as failure, we can solve the prcblem of non-termination.

How should this upper limit be established? It is introduced to make desirable processes _appear_ desirable; it would seem logical to fix it so as to make the process appear as desirable as possible. We shall derive a condition which mimimizes the (un)worthiness of the process, t/p.

Suppose a process has probability of success p, failure q, and non-termination r. Let the cost distribution be described by density function $f(x)$. This function will be "improper" in the sense that it incorporates a Dirac delta of magnitude r at infinity. So,

$$\lim_{T \longrightarrow \infty} \int_0^T f(x)\,dx = p+q$$

but

$$\int_0^{\infty} f(x)\,dx = 1$$

Now, let p_T be the probability that the process succeeds with cost less than T; similarly define q_T. Now, <u>assume</u> that the cost distribution is independent of success or failure. Then

$$p_T = pP[x<T] = pF(T)$$

where

$$F(T) = \int_0^T f(x)\,dx$$

Now, let us choose T to maximize p_T/t_T, where t_T is the expected cost, given that the cost is less than T:

$$t_T = \frac{\int_0^T xf(x)\,dx}{F(T)} = E[t \mid t<T]$$

To maximize p_T/t_T, we solve $(d/dT)(p_T/t_T) = 0$

$$\frac{d}{dT}\left[p_T\right] = p\frac{d}{dT}\int_0^T f(x)\,dx = pf(T)$$

$$\frac{d}{dT}\left[t_T\right] = \frac{F(T)\frac{d}{dT}\int_0^T xf(x)\,dx - [\int_0^T xf(x)\,dx]\frac{d}{dT}\int_0^T f(x)\,dx}{[F(T)]^2}$$

$$= \frac{F(T)Tf(T) - F(T)t_T f(T)}{[F(T)]^2}$$

$$= \frac{Tf(T) - t_T f(T)}{F(T)}$$

Now

$$\frac{d}{dT}\left[\frac{p_T}{t_T}\right] = \frac{t_T \frac{d}{dT}\left[p_T\right] - p_T \frac{d}{dT}\left[t_T\right]}{t^2_T}$$

$$= \frac{t_T pf(T) - p[Tf(T) - t_T f(T)]}{t^2_T}$$

$$= \frac{pf(T)}{t^2_T}[2t_T - T]$$

This is zero iff $T = 2t_T$. Now as $T \longrightarrow \infty$, t_T remains finite, so there will be some X such that $T > 2t_T$ for all $T \geq X$, so that p_T/t_T decreases for $T \geq X$. Also, if there are values Y which satisfy $Y = 2t_Y$, then the largest value must be a maximum (or inflection point).

The behavior of p_T/t_T at $T = 2t_T$ can be determined by evaluating $(d/dT)(2t_T - T)$ since $[pf(T)]/t^2_T > 0$.

$$\frac{d}{dT}(2t_T - T) = \frac{2f(T)}{F(T)}(T - t_T) - 1$$

At $t_T = .5T$,

$$= \frac{Tf(T)}{F(T)} - 1$$

This is less than zero, ie. p_T/t_T is a maximum, if $Tf(T) < F(T)$, that is, if $f(T)$ is less than the uniform density function having the same $F(T)$.

The same threshold can be applied to a discontinuous distribution, with the understanding that T can be less than $2t_T$ if there is zero probability that $T \leq x \leq 2t_T$. That is,

$$T = \min[\, 2t_T \, , \inf\{t : F(t) = p+q\} \,]$$

would be adequate.

There are cases where no sensible solution to $T = 2t_T$ exists. For example, with the exponential distribution $f(x) = ge^{-gx}$, $x \geq 0$, the only solution is T=0 (In fact, T=0 for any monotonic decreasing distribution function). However, this corresponds to our intuition in this case: the threshold should be "small", after which we reconsider, retaining the option of continuing with the remaining process (which will still be exponential).

This points up a feature of the cutoff threshold deserving comment. We introduce it only to make a process

appear as desirable as is reasonable; this does not mean that we are comitted to terminating the process if it exceeds the threshold. All we must do to justify the appearance is reconsider the process when we reach the threshold.

The cutoff $T = 2t_T$ was derived assuming the cost distribution was independent of success or failure. With practical planning processes, this is often not the case. For example, many processes fail only after they have tried all possibilities for success, that is, low cost implies success. However, this is the same as the case analyzed, with q=0. So, it seems the result is stronger than the case from which it was derived. While more complicated cases resist easy solution, the method still should serve as a reasonable approximation.

CHAPTER 4

A STRIPS WORLD APPLICATION

4.1 General description

My hand-simulated STRIPS-type robot uses the world model and operators described in the original STRIPS paper [5], but operates in the style of LAWALY. Assertions are stored in predicate infix form, for example (Door1 Connects Room1 Room5) has the form (Argument1 Predicate Argument2 Argument3). Operators follow STRIPS closely, and are given in functional notation, with a Precondition statement, a Delete list, and an Add list. These specifications are described in detail in the Appendix.

Considerable simplification of the general proposal of Section 3 above was necessary to enable a manageable hand simulation. The principle simplification is to use probability estimates uniformly equal 1.0. It turns out then that the two methods of backing up over OR nodes (Section 3.5.1) are identical. Also, the technique developed in Section 3.5.2 for scheduling ANDed subgoals was not implemented; these are simply scheduled left to right.

Given a goal, the system begins with the left-most literal, by checking its truth against the world model. If it is not currently true, it becomes an OR node with all

operators adding the literal's predicate as the disjuncts. To each operator is attached its precondition as a subgoal; the system then continues by attacking the left-most literal of the precondition in the manner just described. This back chaining continues recursively until all the bottom level goals are true in the world model. Choices along the way are made by the decision criterion of Section 3 above. This includes both choices at OR nodes and the instantiation of free variables, which are treated as the OR of their possible instantiations. Free variables are instantiated left to right. For example, the goal ($X Nextto $Y) first becomes

```
($X N $Y)OR____($X N $Y)
           |
           |
   ($X1 N $Y)
```

The alternative is implicitly restricted to $X \neq X1$.

A stage of elaboration consists of expanding the plan until a new free variable is instantiated.

To develop the cost estimates for goals, let us first consider the simple goal (I Nextto B1). Clearly, the cost of achieving this will be a function of the number of doors on the route from the location of the robot to the box. If they are in the same room, the cost is easily determined, since the plan is known:

```
(I N B1)OR___G2(B1)OR___AND
                         |
         -----------------------------------
         |               |                  |
        (F)        (B1 IN $X)         (I IN R1)
         .               .                  .
         .               .                  .
       true        (B1 IN R1)         (I IN R1)
```

Now, we need a numerical cost function. Suppose we assign unit cost to each of the following planning operations:

1. Performing a match of a literal against the world model. Eg. Testing the truth of (B1 IN $X) to get (B1 IN R1)

2. Attaching a node in the planning tree.

For execution cost, assign "a" units for each operator application. These are admittedly simple cost criteria, but the system and examples described do not depend on this simplicity; it is only a matter of convenience. It has the virtue that no unbounded operation can get away free. The above plan then will cost (8+a), assuming the goal node to be given without charge to the planner.

Consider now the goal (I IN R). Clearly the cost will depend on the room the robot is in, and the target room. We will use the notation r(X) to refer to the room satisfying (X IN r(X)). Let us also define j(X,Y) to be the minimum number of doors on a path joining X and Y. We expect the cost of goal (I IN R) to depend mainly on j(I,R), and will develop a cost estimate based on this assumption. Note that thus the robot effectively has direct access to the information about the routes between

any two rooms, so we should not be surprised that it can avoid false steps in choosing a route. However, this is not unreasonable in an environment familiar to the robot; Siklossy also uses a similar direct access of routes with LAWALY [10].

If j(I,R) = 0, then R=r(I) and (I IN R) is true in the initial model. The goal (I IN R) then has cost 0 (Once it's been placed in the planning tree and evaluated).

Consider now the given world with goal (I IN R5); since r(I) = R1, j(I,R5) = 1. Planning proceeds:

```
(I IN R5)OR___GD($D $R R5)OR___AND
               ____________________|_____
              |         |        |          |
       ($D C $R R5)    (F)   (I IN *R)   (I N *D)
```

When we bind variables, we back up the substitutions and add an alternative:

```
(I IN R5)OR___(I IN R5)
          |
          |
   GD(D1 R1 R5)OR_____AND
                 _______|________________
                |       |        |          |
         (D1 C R1 R5)  (F)   (I IN R1)   (I N D1)
```

Note that the alternative is added to the nearest ancestor for which there is an evaluation; thus the GD operator level is bypassed. The cost is 7 for the new nodes added and 4 for matching, for a total of 11.

The remaining cost via D1 is now only the cost of (I

N D1) plus a, since the others are already true. From the Appendix, (I N D1) is characterized by

$$t = 16+a \qquad f = 1.0(2a)$$
$$s = 16$$

So, the remaining cost via D1 is 16+2a and the GD(D1 R1 R5) branch is characterized by

$$t = 16+2a \qquad f = 1.0(2a)$$
$$s = 16$$

How shall we characterize (I IN R5)? Suppose we wish to search the rooms connected to R5 in sequence until we find a particular R for which $j(I,R) = j(I,R5)-1 = 0$. The expected number of searches will depend on the probability of success on each search; assume this is 0.5. Then we expect 2 searches, after which the remaining cost will be exactly the 16+2a obtained above. We have also shown that each search costs 11. The expected cost for (I IN R5) then is 2*11+16+2a = 38+2a, and the goal can be characterized

$$t = 38+2a \qquad f = \begin{cases} .5(16+2a) \\ .5(38+2a) \end{cases}$$
$$s = 11$$

since if a search fails, the expected remaining cost is the same.

The characterization for $j(I,R) = 2$ can be determined similarly, as can characterizations for the other unary goals; these are tabulated in the Appendix. We note that

this cost structure results in the search behavior we had assumed.

4.2 Example: ($X N $Y)

Backing up of characterizations and re-evaluating choices is illustrated by the goal ($X N $Y), "achieve one box next to another". Let us trace through its planning in detail, for the initial world of Figure 19; the planning tree is shown in Figure 20.

First, the planning tree is initialized to just

```
1($X N $Y)
```

We use numbers before nodes to indicate the order in which they are generated. This goal is matched against the world model; the match fails, so planning begins:

```
1($X N $Y)OR---3($X N $Y)
            |
            |
       2(B1 N $Y)
```

Now there is a choice between expanding 2 and 3; their characterizations are

```
C2:   s = 2           f = .2(143+6a); .8(152+6a)
      t = (153+6a)

C3:   s = 2           f = .2(63+2a); .8(73+2a)
      t = (73+2a)
```

Costs of starting with each are:

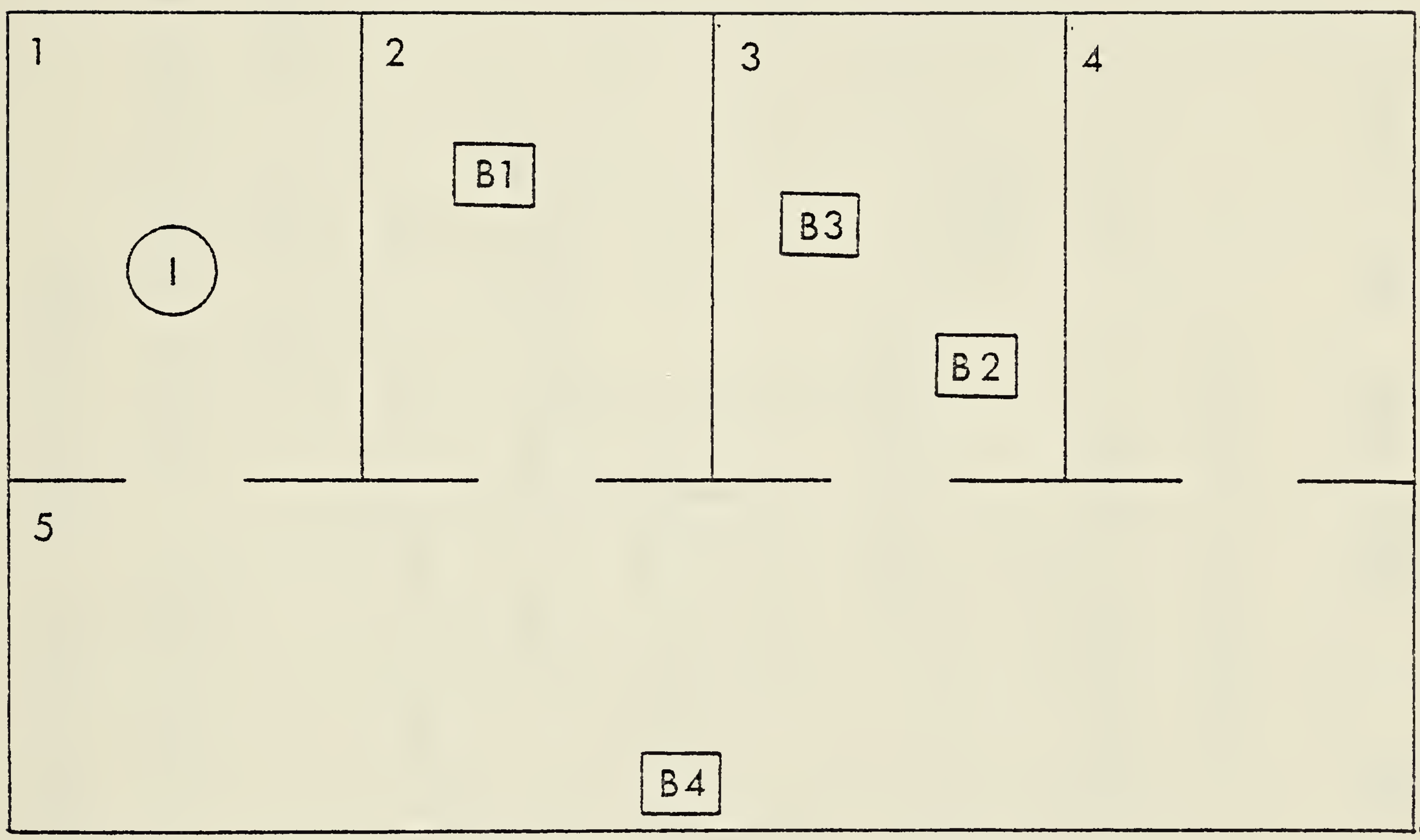


FIG. 19 INITIAL WORLD FOR ($X N $Y)

```
1($X N $Y)OR__3($X N $Y)OR__5($X N $Y)OR__7($X N $Y)OR
          |             |             |             |
          |             |        6(B3 N $Y)    8(B4 N $Y)OR__10(B4 N $Y)OR__12(B4 N $Y)OR
          |             |                                |              |              |
          |             |                           9(B4 N B1)    11(B4 N B2)    13(B4 N B3)
          |             |
          |        4(B2 N $Y)OR__20(B2 N $Y)OR__22(B2 N $Y)
          |                    |              |
          |               19(B2 N B1)    21(B2 N B3)OR__24P(B3 B2)
          |                                               |
   2(B1 N $Y)OR__15(B1 N $Y)OR__17(B1 N $Y)OR             |
             |               |               |            |
       14(B1 N B2)     16(B1 N B3)     18(B1 N B4)        |
                                                          |
                                                    23P(B2 B3)OR
                          ------------------------------|------------------
                          |                                  |           |
                        25AND                              26AND       27AND
         -------------------|------------------              |           |
         |      |      |         |            |              |           |
    28(B2 P)29(F)      |    31(B3 IN R3)32(B2 IN R3)       33-37       38-42
                       |
                 30(I N B2)OR . . . continued overleaf
```

FIG. 20 TREE FOR ($X N $Y)

```
30(I N B2)OR__43G2(B2)OR
               --------+-----------------------------
               |                          |         |
           44AND                       45AND     46AND
    -----------+-------                   |         |
    |       |          |                  |         |
47(F)48(B2 IN R3)49(I IN R3)OR          50-52     53-55
                             |
                             |
                   56GD(D3 R5 R3)OR__57AND
          -----------------------------+--
         |        |     |                |
58(D3 C R5 R3)59(F)     |          61(I N D3)OR__81G2(D3)OR
                        |                                 +-------------------
                        |                                |          |         |
                60(I IN R5)OR__62GD(D1 R1 R5)OR       82AND         |      84AND
                              ---------------|           |          |         |
                             |                           |          |         |
                          63AND                        85-87    83AND       91-93
          -------------------+------                                +--------------------
         |        |       |         |                              |           |        |
64(D1 C R1 R5)65(F)66(I IN R1)67(I N D1)OR__68G2(D1)OR       88(D3 C R5 R3)89(F)90(I IN R5)
                -------------------------------------+-------------
                |                          |                       |
            69AND                       70AND                  71AND
                |                   -------+---------              |
                |                  |        |       |              |
              72-74        75(D1 C R1 R5)76(F)77(I IN R1)       78-80
```

FIG. 20 Conc.

$$c2 = 2+1.0(73+2a) = (75+2a)$$

$$c3 = 2+1.0[.2(63+2a)+.8(73+2a)] = (73+2a)$$

So 3 is chosen. The characterization of node 1 is updated to the backed up OR (= C3):

C1: s = 2 f = .2(63+2a); .8(73+2a)
t = (73+2a)

A like process continues until 8 nodes have been created, that is, all possible boxes have been considered for X. At this point, C2 = C4 = C6 and C8 is different:

C8: s = 2 f = .2(98+4a); .8(108+4a)
t = 108+4a

and by backing up, C8--> C7--> C5--> C3--> C1, since 7 is chosen over 6, 5 over 4, and 3 over 2. Our characterization of node 1 is altered for the first time.

Now (B4 N $Y) is expanded; the alternative is always chosen so that after 13 nodes have been created, C12 = C13:

C12,C13: s = 64 f = .5(90+6a); .5(112+6a)
t = (165+6a)

and C12--> C10--> C8, so our characterization of node 8 changes. This in turn causes backing up to proceed further; now C8--> C7 and node 6 becomes preferred to 7, so C6--> C5 = C4 = C3 = C2 = C1. That is,

C1: s = 2 f = .2(143+6a); .8(153+6a)
t = (153+6a)

Between identical subgoals, the first is chosen, so now (B1 N $Y) is expanded.

Since the instances are all bad, all remaining possibilities for Y are considered, until 18 nodes have been created. Then

C18: s = 64 f = .5(90+6a); .5(112+6a)
t = (165+6a)

and C18--> C17--> C15--> C2, since C14 = C16 = C18.

Now node 3 is chosen over node 2, and since 4 and 5 are equivalent, 4 is expanded.

When 22 nodes have been created,

C22: s = 2 f = .2(143+6a); .8(153+6a)
t = (154+6a)

C21: s = 64 f = .5(68+6a); .5(90+6a)
t = (143+6a)

so 21 is chosen, and the characterization backs up to 20, 4, 3 and 1. The choice path is still to node 21, so the fully instantiated (at this level) subgoal (B2 N B3) is expanded.

Elaboration continues in straightforward fashion until after 93 nodes have been created, the plan

```
Goto2(Door1)
GothruDoor(Door1 from Room1 into Room5)
Goto2(Door3)
GothruDoor(Door3 from Room5 into Room3)
Goto2(Box2)
Pushto(Box2 to Box3)
```

is produced.

This example shows how my proposal can abandon choices as soon as they become unattractive, without planning them to completion. For example, the choice to plan (B4 N $Y) was abandoned before a complete plan was formed. This is in contrast to most robot planning systems, which employ only failure-driven back up.

While this in itself is an achievement, it is no better than the performance of MULTIPLE or other simple evaluation function methods. The need for a look-ahead feature only becomes clear in the next example.

<u>4.3</u> Example: (Q ST ON)

Let us now look at the STRIPS version of the monkey and bananas problem: turning on a light, which requires using a box to reach the switch. We use the initial world model of Figure 21. The goal elaborates as in Figure 22.

After 22 nodes have been created, node 22 (like nodes 1, 8 and 15) is characterized

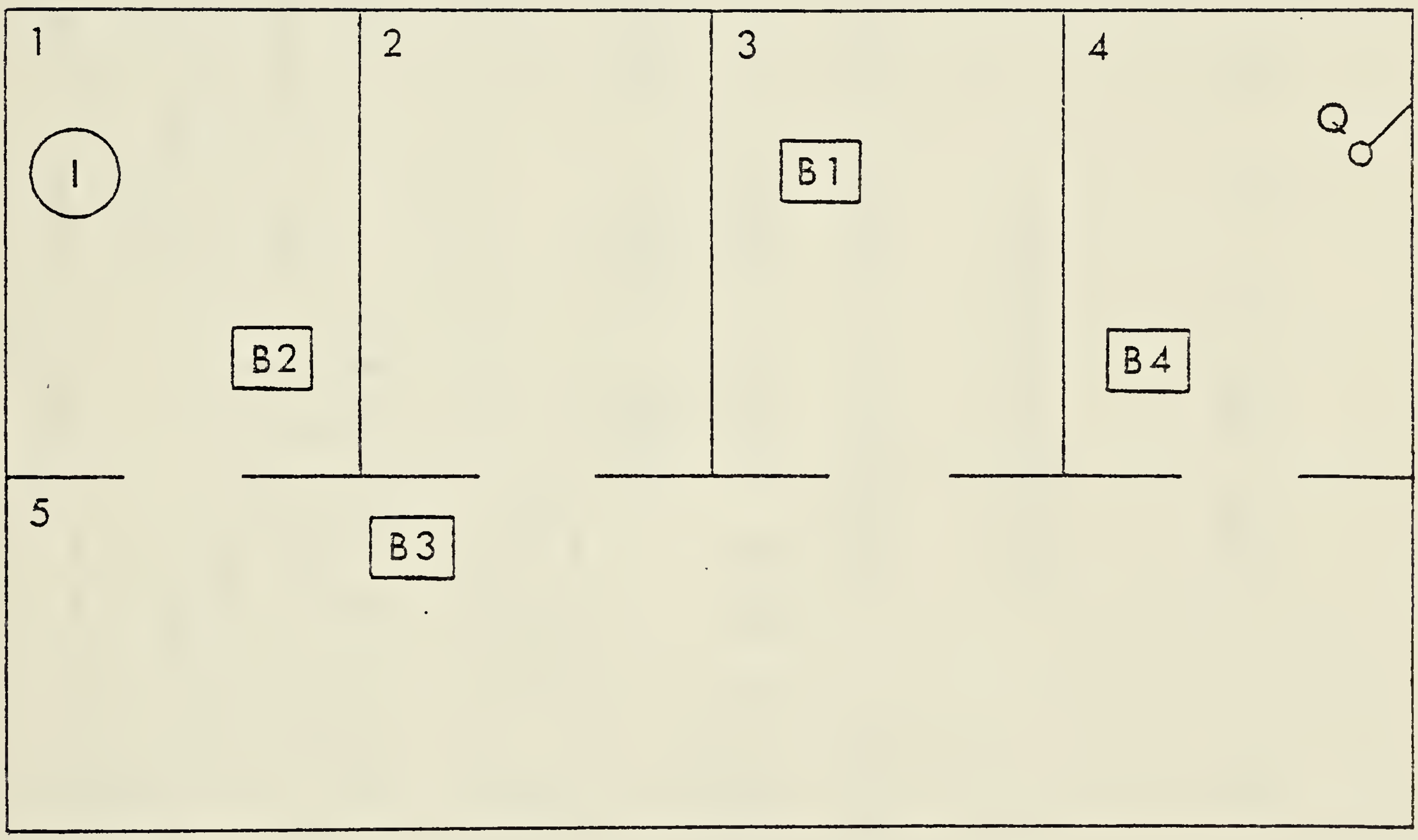


FIG. 21 INITIAL WORLD FOR (Q ST ON)

```
1(Q ST ON)OR_______8(Q ST ON)OR______________15(Q ST ON)OR____22(Q ST ON)
          |                  |                           |
          |                  |                           |
     2T(Q)OR__3AND      9T(Q)OR__10AND              11T(Q)OR__17AND
   _____________|__      ______|__________          ______|_____________
   |    |    |    |      |    |    |     |          |     |     |     |
4(Q T S) | 6(B1 N Q) |  11(Q T S) | 13(B2 N Q) |   18(Q T S)19(B3 T B) | 21(I ON B3)OR
        |         |              |          |                      |            |
   5(B1 T B) 7(I ON B1)     12(B2 T B) 14(I ON B2)          20(B3 N Q)OR        |
                                                                   |    103CU(B3)OR
                  _________________________________________________|       ____|
                  |               |                                        |
             23P(B3 Q)OR     24P(Q B3)                                  104AND
              _______|_____________________                  ______________|_
             |                        |     |                |      |      |
           25AND                    26AND  27AND      105(B3 T B)106(F)107(I N B3)
   _____________|_____________         |     |
   |    |    |    |          |         |     |
28(B3 P)29(F) | 31(Q IN R4)32(B3 IN R4)OR   33-37  38-42
              |                     |
              |                     |
     30(I N B3)OR__43G2(B3)OR   75PD(B3 D4 R5 R4)OR__76AND
                         .                          .
                         .                          .
                         .                          .
                     continued overleaf
```

FIG. 22 TREE FOR (Q ST ON)

continued overleaf

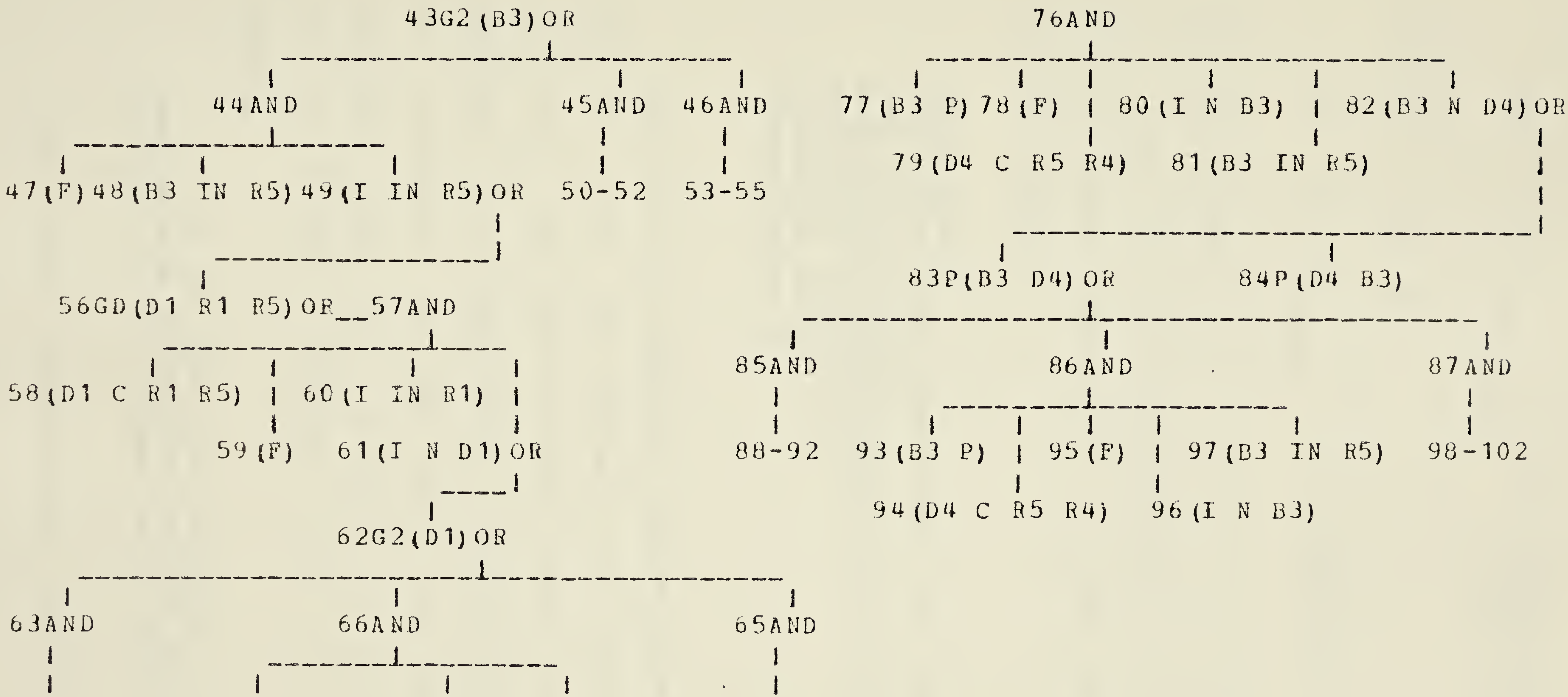

43G2(B3)OR
44AND
45AND
46AND
47(F)
48(B3 IN R5)
49(I IN R5)OR
50-52
53-55
56GD(D1 R1 R5)OR__57AND
58(D1 C R1 R5)
59(F)
60(I IN R1)
61(I N D1)OR
62G2(D1)OR
63AND
66AND
65AND
76AND
77(B3 P)
78(F)
79(D4 C R5 R4)
80(I N B3)
81(B3 IN R5)
82(B3 N D4)OR
83P(B3 D4)OR
84P(D4 B3)
85AND
86AND
87AND
88-92
93(B3 P)
94(D4 C R5 R4)
95(F)
96(I N B3)
97(B3 IN R5)
98-102

FIG. 22 Conc.

```
C22:    s = 11              f = .2(152+8a); .2(174+8a);
        t = (190.5+8a)          .6(190.5+8a)
```

and node 16 will be

```
C16    s = 64              f = .5(99+8a), .5(121+8a)
       t = (174+8a)
```

The clear choice is 16, and this will back up to node 1 so C1 = C8 = C15 = C16. The remainder of the elaboration is straightforward until after 107 nodes have been created the plan emerges:

```
Goto2(Door1)
GothruDoor(Door1 from Room1 into Room5)
Push(Box3 to Door4)
PushthruDoor(Box3 through Door4 from Room5 into Room4)
Push(Box3 to switch Q)
ClimbUp(on Box3)
Turnon(switch Q)
```

There are two important observations about this plan. First is that the optimal plan involving Box4 is not considered. Once Box3 has been found, the system does not "believe" the added effort to search for a better box to be worthwhile. In this case the "belief" is wrong: expanding node 22 at a cost of 11 will reduce remaining cost by (174+81) - (152+8a) = 22. Of course, to quote Feldman,

> . . . if we use the expected utility as a measure when searching for good plans, we do not guarantee good outcomes, only good strategies.

A more subtle observation, however, is this: the

search continued after Box2 had been tried (node 14), even though the expected cost with Box2 is <u>less than the expected cost of "try another box".</u>

$$(196+8a) < .2(174+8a) + .2(152+8a) + .2(196+8a) + .4(286+12a) = (218.8+9.6a)$$

This illustrates the advantage to be gained from one level of look-ahead in our characterization of alternatives: a smart search strategy must do better than merely picking at random (the result with a simple characterization).

CHAPTER 5

CONCLUSIONS

This thesis addresses the problem of optimizing the cost of a plan, including both the planning and execution costs. An explicit proposal for doing this has been developed, based on classical statistical decision theory. The result is an evaluation function approach in the style of MULTIPLE or A*, but the evaluation is in terms of changes in expected cost and probability, rather than just probability or just expected cost. This refinement was introduced after simple evaluation functions were argued to be inadequate (Section 3.2). While my system can be considered as a generalization of MULTIPLE and A*, it employs totally different backing up methods from MULTIPLE; thus it has no "special case" which reduces it to MULTIPLE. It does, however, reduce to an A* type heuristic on remaining cost if probabilities are all 1.0 and the cost distributions are not distributed at all, but consist only of a single point.

When one attempts to include the cost of analysis in a decision analysis, one seems led to an infinite regress: what is the cost of analyzing the cost of analysis? In practice, one need never pose this question; one merely supplies cost estimates for processes which include the cost of analysis. Thus the analysis necessary to choose

among processes (the computations of Section 3.3 - 3.5 above) is included as an "overhead" in the individual processes.

This avoids the question of how these overhead costs might be determined, or of how they are finite if they hide an infinite regress. Firstly, they are finite because the proposed decision criterion limits the regress to one step. Secondly, they can be determined by merely computing the expected number of times the continued pursuit of the process will be reconsidered, since each reconsideration involves fixed, finite cost. Even more directly, such cost estimates could be acquired automatically by recording observed costs of a process for various situations. The fact that performance, and therefore cost, is dependent on the cost estimates themselves, should result in a learning process which propagates information by iterative refinement of cost estimates. This hope is strengthened by the observation that the dependency seems to be one way; estimates for complicated processes or situations depend on the performance with simpler ones. Thus, look-ahead cost estimates seem much easier to acquire than one might think; the "infinite regress" seems more a conceptual problem than a real one.

The decision proposal developed here has a number of weaknesses, which seem a necessary evil in developing it at all.

First is the problem which emerged when backing up characterizations of alternatives at an OR node (Section 3.5.1). Using the logically developed method, it emerged that (A OR B) could be a worse alternative than A alone, a logical contradiction. The solution adopted was an ad hoc "patch up", at the expense of logical consistency. This difficulty is rooted in the assumption that there is no alternative to (A OR B); this assumption is usually not valid. Removing it, however, is not easy, for then the evaluation of any single alternative depends on every other alternative in the planning tree. This is what we were trying to avoid in the first place with only one level of look-ahead. Despite much deliberation, there seems to be no easy way out.

Second is the handling of backing up at AND nodes. Here, we were forced to devise an ordering of the subgoals so that the first step of one subgoal could become the first step of the AND. This was dictated by the representation of the distributions; if distributions were represented by their moments, the computational problem of convolving them (Section 3.5.2) would disappear, and we could let the first step of the AND be the combined first steps of the subgoals. This would be more natural if the subgoals were similarly characterized, as we expect within one level of hierarchical planning.

Thus we see that there are other possibilities for

the development of a look-ahead planner than those explored here. While these were bypassed at the time, we should now (like the planner we are trying to develop) reconsider these choices. What should be done differently, if this work were to be done again? I believe that logical consistency is a greater goal than some of the choices made would reflect; this would dictate making realistic assumptions about alternatives available when computing costs. A method should be sought to make this computationally feasible. If I could offer a more concrete suggestion as to how this could be achieved, I surely would. However, I can offer the following speculations.

Having established the need for characterization of a process by both cost and probability in Section 3.3.1, I would like to offer some hope of relaxing this complication. Probability estimates were introduced to avoid infinite expected cost when failure was a possibility. However, as we then saw in Section 3.3.3, we used these cost estimates in such a way that the infinities cancelled out. Could it be that, following the lead of quantum mechanics, we could simply ignore the troublesome infinities? We could then treat failure merely as the expected cost of continuation becoming infinite, and characterize processes by a simple one dimensional probability distribution.

This is not so easy as it sounds; the conceptual

hurdle of floating infinities is high. However, the situation that gave rise to it may be artificial. The situation we have considered involves a top level goal with no alternatives, but which admits the possibility of failure. Realistically, a goal like (Box1 Nextto Box2) never has such a high priority. Living systems always possess an alternative, and have a top level goal like "Do the best you can"; such a goal cannot fail. Local "failure" then is not infinitely costly when there is a global alternative. Stating this another way, subgoals must be assigned realistic benefits (inverse costs), rather than the implicit infinite benefit of a goal without alternatives.

This would lead to a change in our assumed strategy for a goal like (A OR B); it would no longer be either "try A, if that fails, try B", or "try B, if that fails, try A", because the cost might outweigh the benefit. The result would be a changed decision criterion and a different way of backing up distributions, hopefully a change enabling logical consistency.

Despite these deficiencies, a powerful new decision maker for use in planning has been proposed, one which takes into account the cost of planning. It should be noted in this regard that planning and execution are treated as merely two types of the same activity, and so the techniques developed here are useful for execution

monitoring as well as guiding planning. The significant difference is one of reversability; when planning, one can simply return to a bypassed choice with no cost, whereas with execution, one has to pay to re-establish a previous state of the world (if one can do it at all). However, this is just another instance of interdependence among the costs of alternatives; ignoring it should be no more detrimental than ignoring other interdependencies.

We gain insight into the power of the look-ahead planning proposal by considering it as a meta-planner, choosing among planning processes. With a simple evaluation function, the meta-planner intervenes only once at the start of planning, to select a particular planning process to achieve the goal; any subsequent reconsideration is inconsistent with a simple evaluation function (see Section 3.2 above). The proposed meta-planner, however, "knows" it will intervene again; the decisions it makes depend on this knowledge. Thus, it is in some sense "self aware". Indeed, this is a reversal of the feedback approach usual in process control; rather, it is "feed forward". That is, the proposed meta-planner not only corrects its past errors, but anticipates its current ones. This self awareness or anticipation of self is a much lauded feature of human intelligence. Incorporating it into a planning system as done here thus seems like a valuable step on the road to artificial intelligence.

REFERENCES

[1] Coles, S. L., Robb, A. M., Sinclair, P. L., Smith, M. H., Sobek, R. R. "Decision analysis for an experimental robot with unreliable sensors". Proc. IJCAI 4 (1975), 749-757

[2] DeGroot, M. H. Optimal Statistical Decisions. McGraw-Hill, New York, 1970

[3] Feldman, J. A., and Sproull, R. F. "Decision theory and artificial intelligence II: The hungry monkey". Technical report no. 2, Computer Science Dept., University of Rochester, New York (Nov. 1974)

[4] Feldman, J. A., and Yakimovsky, Y. "Decision theory and artificial intelligence: I. A semantics-based region analyzer". AI 5 (1974), 349-371

[5] Fikes, R. E., and Nilsson, N. J. "STRIPS: A new approach to the application of theorem proving to problem solving". AI 2 (1971), 189-208

[6] Jacobs, W., and Kiefer, M. "Robot decisions based on maximizing utility". Proc. IJCAI 3 (1973), 402-411

[7] Nilsson, N. J. Problem-solving methods in artificial intelligence. McGraw-Hill, 1971

[9] Sacerdoti, E. D. A structure for plans and behavior. American Elsevier, New York, 1977

[8] Sacerdoti, E. D. "Planning in a hierarchy of abstraction spaces". AI 5 (1974), 115-135

[10] Siklossy, L., and Dreussi, J. "An efficient robot planner which generates its own procedures". Proc. IJCAI 3 (1973), 423-430

[11] Simon, H. A., and Kadane, J. B. "Optimal problem-solving search: all-or-none solutions". AI 6 (1975), 235-247

[12] Slagle, J. R., and Bursky, P. "Multipurpose theorem-proving heuristic program". JACM 15#1 (1968), 85-99

APPENDIX I

ROBOT SPECIFICATIONS

Conventions

-All assertions and operators may contain lower case letters; these are for readability only. Thus (Door1 Connects Room1 Room5) is an explainatory elaboration of (D 1 C R1 R5)

Assertions are written in predicate infix notation

(Argument1 Predicate Argument2 Arguement3 ...)

-Operators are written in functional notation:

Operator-name (Argument1 Arguement2 ...)

-Arguments and predicates are delimited by blanks (and lower case letters)

-The relevance of an operator to a unit (sub)goal is determined by matching the literals marked with # on its add list with the goal literal. Matching of the operator arguments is assumed to be restricted in the obvious way (eg. the argument of Goto1 is restricted to locations, the arguments of Pushto are restricted not to be the robot, etc.)

-Elaboration of conjuncts proceeds left to right, in the manner of PLANNER or LAWALY. That is, in planning Pushto(X,Y), the system will fully elaborate a plan for (I Nexto X) before elaborating the subgoal (X IN r(Y)) any further. Also, execution of the leftmost subgoals is modelled in the manner of STRIPS; the

result is that the input world model to any subgoal is fully determined before that subgoal is elaborated.

-Once an OR node has been expanded, a choice is made using the proposed decision criterion. This may require further expansion to enable the characterizations of the ORed alternatives to be evaluated. For example, in Figure <tree for (Q ST ON)>, nodes 3 through 7 must be added to evaluate node 2.

-A unit (sub)goal is treated as an OR node; its successors are the relevant operators resulting from add list matching.

-Free variables are denoted by $Variable-name. These are bound by matching the first literal (unit subgoal) involving them with the world model. If an alternative instanstiation is possible, an alternative (sub)goal is sprouted from the nearest ancestor node for which there is a tabulated evaluation function.

An illustration of the preceeding 2 rules is from Figure <tree for (Q ST ON)>. Node 1 starts as

```
1(Q ST ON)OR
          |
          |
     2 T(Q)OR
```

since there is only one match for (Q ST ON) in the operator add lists. No choice is required of the unary OR. When node 3 is expanded, a free variable $X

is introduced in node 5, the subgoal ($X Type Box). Matching produces the instantiation Box1 for X and the availability of alternative boxes. Nodes 3 and 2 have no tabulated evaluation functions, but node 1 does, so the alternative node 8 is sprouted.

-Free variables generated together are instantiated left to right. The ($X Nexto $Y) example (Section 4.1) illustrates this.

<u>Initial world model</u>

```
(Door1 Connects Room1 Room5)   (Door1 Connects Room5 Room1)
(Door2 Connects Room2 Room5)   (Door2 Connects Room5 Room2)
(Door3 Connects Room3 Room5)   (Door3 Connects Room5 Room3)
(Door4 Connects Room4 Room5)   (Door4 Connects Room5 Room4)

(Box1 AT AA)        (Box2 AT BB)
(Box3 AT CC)        (Switch1 AT DD)

(Box1 IN Room1)        (Box2 IN Room1)
(Box3 IN Room1)        (I IN Room1)
(Switch1 IN Room1)

(Box1 Type Box)        (Box2 Type Box)
(Box3 Type Box)        (Door4 Type Door)
(Door3 Type Door)        (Door2 Type Door)
(Door1 Type Door)        (Switch1 Type Switch)

(Box1 Pushable)        (Box2 Pushable)
(Box3 Pushable)

(FF LOCated in Room4)
(EE ATR)
(onFloor)
(Switch1 STatus OFF)
```

<u>Operators</u>

```
Goto1(M)  Robot goes to location M
     Preconditions:
          (M LOC $R) (F) (I IN R)
```

```
     Delete:  ($ ATR) (I N $)
     Add:  #(M ATR)

Goto2(X)  Robot goes next to item X
     Preconditions:
          (F) (X IN $R) (I IN R)
       OR (X C $R $T) (F) (I IN R)
       OR (X C $R $T) (F) (I IN T)
     Delete:  ($ ATR) (I N $)
     Add:  #(I N X)

Pushto(X,Y)  Robot pushes object X next to item Y
     Preconditions:
          (X P) (F) (I N X) (Y IN $R) (X IN R)
       OR (X P) (X C $R $T) (F) (I N X) (X IN R)
       OR (X P) (X C $R $T) (F) (I N X) (X IN T)
     Delete:  ($ ATR) (I N $) ($ N X) (X AT $) (X N $)
     Add:  #(X N Y) #(Y N X)
          (I N X)

Turnlight(Q)  Robot turns on lightswitch Q
     Preconditions:
          (Q T S) ($X T B) (X N Q) (I ON X)
     Delete:  (Q ST OFF)
     Add:  #(Q ST ON)

ClimbUponbox(X)  Robot climbs up on box X
     Preconditions:
          (X T B) (F) (I N X)
     Delete:  ($ ATR) (F)
     Add:  #(I ON X)

ClimbDownoffbox(x)  Robot climbs down off box X
     Preconditions:
          (X T B) (I ON X)
     Delete:  (I ON X)
     Add:  #(F)

GothruDoor(D,R,T)  Robot goes through Door from Room
     into room T
     Preconditions:
          (D C R T) (F) (I IN R) (I N D)
     Delete:  ($ ATR) (I N $) (I IN $)
     Add:  #(I IN T)

PushthruDoor(O,D,R,S)  Robot pushes Object through Door
     from Room into room S
     Preconditions:
          (O P) (F) (D C R S) (I N O) (O IN R) (O N D)
     Delete:  (I AT $) (I N $) (O N $) ($ N O) (O AT $)
          (I IN $) (O IN $)
     Add:  #(O IN S)
          (I IN S) (O N D) (I N O)
```

<u>Characterizations of unit subgoals</u>

The following tables give the characterization of selected unit subgoals. For each unit subgoal, several characterizations are provided for various input world situations.

The followint notation is used in describing situations:

1. r(X) the room containing X

2. j(X,Y) the minimum number of doors on a path from r(X) to r(Y).

3. Rooms fall into two categories: firstly, the hall, R5, designated h, and secondly the others, designated u,v,w; different symbols denote distinct rooms. That is, the situation (I IN R1), (BOX1 IN R1) would be designated u,u while (I IN R1), (BOX1 IN R2) would be designated u,v.

4. Characterizations are the expected cost t, initial cost s, and a distribution f(t'). This distribution is always a discrete set of points, and is described by a notation
   f: q1(t1), q2(t2), . . .
   whare value t1 occurs with probability q1, etc.

5. The operator Pushto also has a characterization, to allow P(X Y) and P(Y X) to be distinguished.

(I IN R)

j(I,R)	t	s	f(t')
0	0		
1	45+2a	11	.5(23+2a), .5(45+2a)
2	90+4a	11	.5(68+4a), .5(90+4a)

(I N X)

j(I,X)	t	s	f(t')
0	16+a	16	1.0(a)
1	61+3a	27	.5(23+3a), .5(45+3a)
2	106+5a	27	.5(68+5a), .5(90+5a)

(X IN R)

j(I,X)	j(X,R)	t	s	f(t')
	0	0		
	1	83+3a	15	.5(53+3a), .5(83+3a)
	2	150+5a	15	.5(120+5a), .5(150+5a)
	0	0		
	1	128+5a	15	.5(98+5a), .5(128+5a)
	2	195+7a	15	.5(165+7a), .5(195+7a)
	0	0		
	1	173+7a	15	.5(143+7a), .5(173+7a)
	2	240+9a	15	.5(210+9a), .5(240+9a)

(X N Y)

min j(I,X) J(I,Y)	j(X,Y)	t	s	f(t')
0	0	53+2a	53	1.0(2a)
0	1	120+4a	68	.5(37+4a), .5(67+4a)
0	2	187+6a	68	.5(104+6a), .5(134+6a)
1	0	98+4a	64	.5(23+4a), .5(45+4a)
1	1	165+6a	64	.5(90+6a), .5(112+6a)
1	2	232+8a	64	.5(157+8a), .5(179+8a)
2	0	143+6a	64	.5(68+6a), .5(90+6a)
2	1	210+8a	64	.5(135+8a), .5(157+8a)
2	2	277+10a	64	.5(202+10a, .5(224+10a)

P(X Y)

like above table, but column headings

j(I,X)	j(X,Y)	t	s	f(t')

($X IN R)

R	r(I)	t	s	f(t')
h	h	128+5a	17	.5(98+5a), .5(128+5a)
h	u	91+3a	2	.25(83+3a), .75(91+3a)
u	u	136+5a	2	.25(128+5a), .75(136+5a)
u	h	91+3a	2	.25(83+3a), .75(91+3a)
u	v	136+5a	2	.25(128+5a), .75(136+5a)

($X N Y)

r(Y)	r(I)	t	s	f(t')
h	h	63+2a	2	.2(53+2a), .8(63+2a)
h	u	108+4a	2	.2(98+4a), .8(108+4a)
u	u	63+2a	2	.2(53+2a), .8(63+2a)
u	h	108+4a	2	.2(98+4a), .8(108+4a)
u	v	153+6a	2	.2(143+6a), .8(153+6a)

($X N $Y)

t = 73+2a

s = 2

f(t'): .2(63+2a), .8(73+sa)

(Q ST ON)

r(I)	r(Q)	t	s	f(t')
h	h	117+4a	11	.2(62+4a), .8(117+4a)
h	u	145.5+6a	11	.2(117+6a), .2(129+6a), .6(145.5+6a)
u	h	145.5+6a	11	.2(117+6a), .2(129+6a), .6(145.5+6a)
u	u	117+4a	11	.2(62+4a), .8(117+4a)
u	v	190.5+8a	11	.2(152+8a), .2(174+8a),